MAGAZINE BSD

FOR NOVICE AND ADVANCED USERS

WHAT'S NEW IN PC-BSD 9.1

INSIDE

VOL.6 NO.09
ISSUE 09/2012(38)
1898-9144

FreeNAS 8 POWERED

TrueNAS™

UNIFIED. SCALABLE. FLEXIBLE.

Across all industries the demands of data infrastructure have soared to new heights.

As capacity requirements continue to rise at an ever-increasing rate, performance must not be compromised. The hybrid architecture and advanced software capabilities of the TrueNAS appliance enable users to be more agile, effectively manage the explosion of unstructured data and deploy a centralized information storage infrastructure. Whether it's backing virtual machines, business applications, or web services, there's a TrueNAS appliance suited to the task.

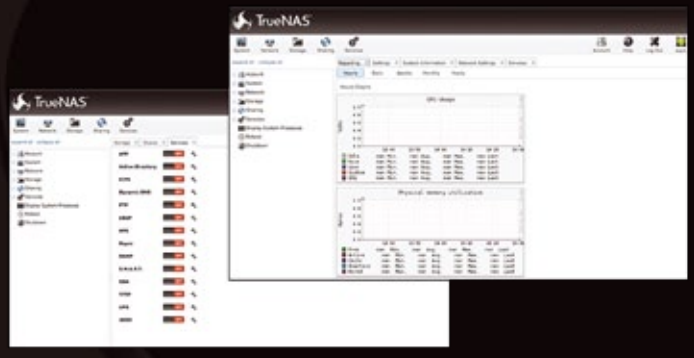
TrueNAS™ Storage Appliances: Harness The Cloud

iXsystems' TrueNAS Appliances offer scalable high-throughput, low latency storage

All TrueNAS Storage Appliances feature the Intel® Xeon® Processors 5600 series, powering the fastest data transfer speeds and lowest latency possible. TrueNAS appliances come in three lines: Performance, Archiver, & High Availability. High-performance, high-capacity ioMemory modules from Fusion-io are available in the TrueNAS Enterprise, Ultimate, and Archiver Pro models.

Key Features:

- **One or Two Six-Core Intel® Xeon® Processors 5600 series**
- **Share Data over CIFS, NFS and iSCSI**
- **Hybrid storage pool increases performance and decreases energy footprint**
- **128-bit ZFS file system with up to triple parity software RAID**



*Optional component

FEATURES	TrueNAS Pro	TrueNAS Enterprise	TrueNAS Ultimate	TrueNAS Fileshare	TrueNAS Archiver Pro	TrueNAS Pro-HA	TrueNAS Enterprise-HA	TrueNAS Ultimate-HA
	PERFORMANCE			ARCHIVER		HIGH AVAILABILITY		
Fusion-io Card		X	X		X			
Deduplication					X			
High Availability						X	X	X
Gigabit NICs	Quad	Dual	Dual	Dual	Dual	Quad	Quad	Dual
10 Gigabit NICs		Dual*	Quad*		Dual*			Quad*
Max Main Memory	48Gb	96Gb	192Gb	48Gb	192Gb	48Gb	96Gb	192Gb
Max Capacity	320TB	500TB	450TB	680TB	2.2PB	250TB	310TB	1.4PB
Rack Units	2U/4U	2U/4U	4U	2U/4U	4U	3U	3U	Dual 3U

Call iXsystems toll free or visit our website today!

1-855-GREP-4-IX | www.iXsystems.com

Dear Readers,

The PC-BSD 9.1 will be released this month. We are sure that you just can't wait for it. To sweeten your time till this event, we publish in Dev Corner an article by Dru Lavigne about the new features in PC-BSD 9.1. So, you can check now what you are waiting for.

September's Dev Corner is dedicated to PC-BSD. In the second article from this section Kris Moore will show you how to set up OwnCloud via the Warden. It received very good reviews from betatesters, so you should enjoy it much.

We also introduced a new series by Rob Somerville. This time he will show you step by step how to build a search engine using Apache SOLR. A great grasp of practical knowledge – just as you like it.

This month we launched The Best of BSD 2011. You will find there the best BSD Magazine articles of 2011 with updates. The idea is to sum up the 2011 year, not write a new one, so still you can find the references to old releases. It gives the opportunity to compare the past with the present and to follow the development of BSD systems and users' needs. The other purpose of this issue is to support BSD Magazine, so it can maintain its position on the market as a free on-line magazine.

You may buy the issue on: http://stackmag.org

Wish you a good read!
Patrycja Przybylowicz
& BSD Team

MAGAZINE BSD

Editor in Chief:
Ewa Dudzic
ewa.dudzic@software.com.pl

Supportive Editor
Patrycja Przybyłowicz
patrycja.przybylowicz@software.com.pl

Contributing:
Dru Lavigne, Kris Moore, Rob Somerville, Paul McMath, Michael Shirk, Luca Ferrari, Giovanni Bechis, Jeroen van Nieuwenhuizen

Top Betatesters & Proofreaders:
Paul McMath, Bjørn Michelsen, Barry Grumbine, Babak Farrokhi, Eric Geissinger, Luca Ferrari, Imad Soltani, Mani Kanth, Luiz Claudio Pacheco, Zander Hill, Eric De La Cruz Lugo

Special Thanks:
Dru Lavigne

Art Director:
Ireneusz Pogroszewski

DTP:
Ireneusz Pogroszewski
ireneusz.pogroszewski@software.com.pl

Senior Consultant/Publisher:
Paweł Marciniak *pawel@software.com.pl*

CEO:
Ewa Dudzic
ewa.dudzic@software.com.pl

Production Director:
Andrzej Kuca
andrzej.kuca@software.com.pl

Executive Ad Consultant:
Ewa Dudzic
ewa.dudzic@software.com.pl

Advertising Sales:
Patrycja Przybyłowicz
patrycja.przybylowicz@software.com.pl

Publisher :
Software Press Sp. z o.o. SK
ul. Bokserska 1, 02-682 Warszawa
Poland
worldwide publishing
tel: 1 917 338 36 31
www.bsdmag.org

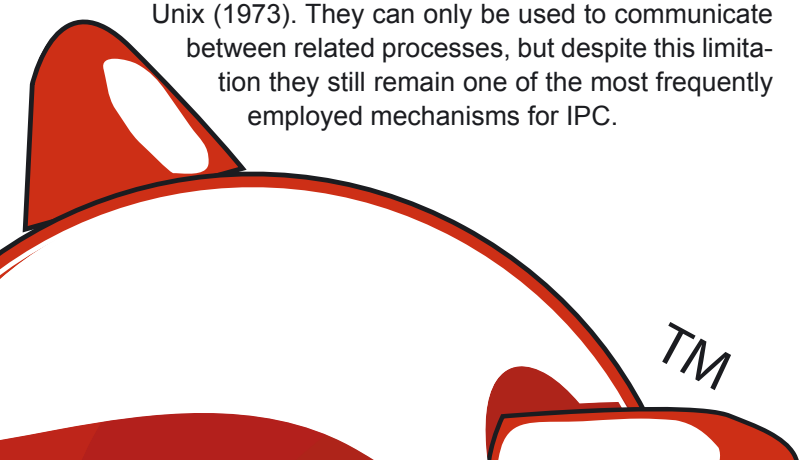
Software Press Sp z o.o. SK is looking for partners from all over the world. If you are interested in cooperation with us, please contact us via e-mail: editors@bsdmag.org

Mathematical formulas created by Design Science MathType™.

Get Started

Developers Corner

How To



Security

Interview

Nmap

The Network Swiss Army Knife

Nmap ("Network Mapper") is a GPL utility for network discovery and security auditing. Many systems and network administrators find it very useful for network inventory, monitoring hosts and services uptime, debugging network related problems, and many other tasks.

What you will learn…

- the basic functionalities of Nmap 6

What you should know…

- basic tcp/ip knowledge

After three years of development Nmap 6 has been released. This release comes with some new features and many improvements.

To install nmap on OpenBSD just run the command `pkg_add -i nmap`.

If you want to install the gui as well: `pkg_add -i nmap-zenmap`.

In OpenBSD nmap has been recently updated to the latest version: 6.01 If you want to test all new

improvements you should install a snapshot or wait for the 5.2 release of OpenBSD.

Nmap is mostly used to check known hosts for open, closed or filtered ports. To do this execute it with just the name of the host you want to scan: Listing 1.

If you want to know more info about your target just add some options: Listing 2.

By adding "-A" option you ask nmap to let you know more informations about the target you are scanning; the "-T4" option increases nmap's speed of execution (one of the improvements of nmap6). Keep in mind that the faster nmap is scanning, the easier it will be for someone to notice, either by seeing the kind of the packets nmap generates as they're travelling on the wire or by noticing degraded performance on the system being scanned.

In Nmap 6.00 there are many more nse scripts than in previous versions; the *Nmap Scripting Engine* (NSE) is one of Nmap's most powerful and flexible features.

It allows users to write simple Lua scripts to automate network tasks including vulnerability detection and exploitation. For example, to test whether our web server has any known vulnerabilities we can run: Listing 3.

Listing 1. *Localhost scan*

```
$ nmap 127.0.0.1

Starting Nmap 6.01 ( http://nmap.org ) at 2012-07-27
                   22:58 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000054s latency).
Not shown: 993 closed ports
PORT     STATE SERVICE
13/tcp   open  daytime
25/tcp   open  smtp
37/tcp   open  time
113/tcp  open  ident
587/tcp  open  submission
631/tcp  open  ipp
6000/tcp open  X11

Nmap done: 1 IP address (1 host up) scanned in 17.34
                    seconds
```

Listing 2. *Service detection scan on localhost*

```
$ nmap -A -T4 127.0.0.1

Starting Nmap 6.01 ( http://nmap.org ) at 2012-07-27 23:01 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000058s latency).
Not shown: 993 closed ports
PORT     STATE SERVICE VERSION
25/tcp   open  smtp    Sendmail 8.14.5/8.14.5
| smtp-commands: bigio.snb.it Hello giovanni@localhost [127.0.0.1], pleased to meet you, ENHANCEDSTATUSCODES,
                  PIPELINING, 8BITMIME, SIZE, DSN, ETRN, DELIVERBY, HELP,
|_ 2.0.0 This is sendmail version 8.14.5 2.0.0 Topics: 2.0.0 HELO EHLO MAIL RCPT DATA 2.0.0 RSET NOOP QUIT HELP VRFY
                  2.0.0 EXPN VERB ETRN DSN AUTH 2.0.0 STARTTLS 2.0.0 For more info use "HELP <topic>". 2.0.0 To
                  report bugs in the implementation see 2.0.0 http://www.sendmail.org/email-addresses.html 2.0.0
                  For local information send email to Postmaster at your site. 2.0.0 End of HELP info
631/tcp  open  ipp     CUPS 1.5
| http-methods: Potentially risky methods: PUT
|_See http://nmap.org/nsedoc/scripts/http-methods.html
| http-robots.txt: 1 disallowed entry
|_/
6000/tcp open  X11     (access denied)
Service Info: Host: scan.test.lan; OS: Unix

Service detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 24.57 seconds
```

Listing 3. *Nse scripts check on an http server*

```
$ nmap -sC -p80 127.0.0.1

Nmap scan report for localhost (127.0.0.1)
Host is up (0.00063s latency).
PORT   STATE SERVICE
80/tcp open  http
|_http-title: Test Page for Apache Installation
|_http-methods: No Allow or Public header in OPTIONS response (status code 405)

Nmap done: 1 IP address (1 host up) scanned in 0.97 seconds
```

Listing 4. *Ipv6 scan*

```
$ nmap -6 -p25 ::1

Starting Nmap 6.01 ( http://nmap.org ) at 2012-07-27 23:14 CEST
Nmap scan report for localhost (::1)
Host is up (0.0022s latency).
PORT   STATE SERVICE
25/tcp open  smtp

Nmap done: 1 IP address (1 host up) scanned in 0.28 seconds
```

Listing 5. *nping testing on localhost*

```
$ sudo nping -c 1 -icmp 127.0.0.1

Starting Nping 0.6.01 ( http://nmap.org/nping ) at 2012-07-27 23:23 CEST
SENT (0.0067s) ICMP 127.0.0.1 > 127.0.0.1 Echo request (type=8/code=0) ttl=64 id=45674 iplen=28
RCVD (0.0075s) ICMP 127.0.0.1 > 127.0.0.1 Echo reply (type=0/code=0) ttl=255 id=47169 iplen=28

Max rtt: 0.527ms | Min rtt: 0.527ms | Avg rtt: 0.527ms
Raw packets sent: 1 (28B) | Rcvd: 1 (28B) | Lost: 0 (0.00%)
Tx time: 0.00181s | Tx bytes/s: 15435.50 | Tx pkts/s: 551.27
Rx time: 1.00729s | Rx bytes/s: 27.80 | Rx pkts/s: 0.99
Nping done: 1 IP address pinged in 1.02 seconds
```

One of the main improvements to Nmap 6.00 is full ipv6 support; all options now

support ipv6 addresses. Just add the "-6" option to your command line: Listing 4.

Some nmap options need root access (for example "-O" parameter used to detect the remote operating system version), but most options works when nmap runs as an unprivileged user as well.

Nping and Ncat

In recent Nmap versions (5.00 and later), a couple of new tools have been added: "nping" and "ncat".

Nping is a tool for network packet generation tool capable using a wide variety of protocols. It can be used for raw packet generation, network response analysis, network stack stress tests, route tracing, and more (Listing 5).

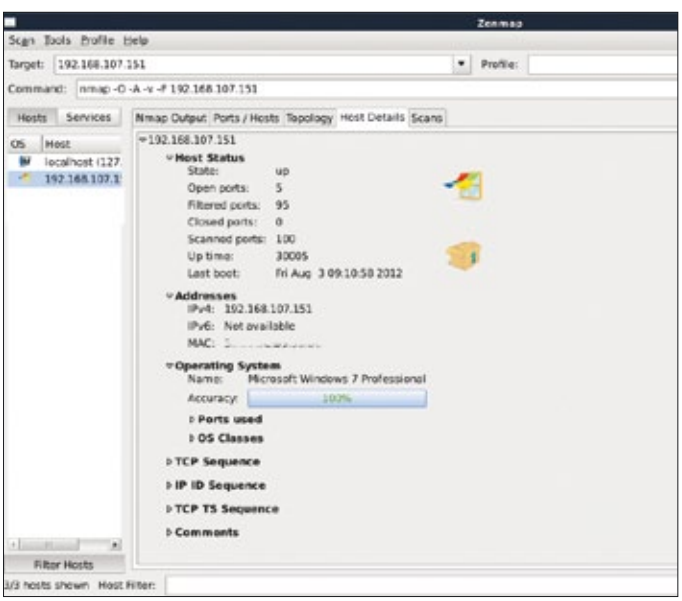


Figure 1. *Service detection scan using nmap gui*

Ncat is a feature-packed networking utility which reads and writes data across networks from the command line. Ncat is designed to be a reliable back-end tool to instantly provide network connectivity to other applications and users.

It is frequently used to create simple proxies for other applications.

For example, to create a simple http proxy server just type the following command line:

```
ncat -l --proxy-type http localhost 8080
```

Zenmap: Nmap for Everybody

nmap also has gui interface named Zenmap; with Zenmap you have all of nmap's options available; you can scan hosts, networks and have all fancy reports you want with just few clicks.

As a plus you can save your scans in an xml config file to repeat it later.

With the new nmap you can probe for open, closed, filtered ports on remote hosts and discover which operating systems remote hosts are running even faster than in previous releases; you can also save your scanning results in many file formats which can be used for post-processing with other tools.

GIOVANNI BECHIS

Giovanni Bechis lives in Italy with his wife and son. He is an OpenBSD developer and the owner of SnB, a software house which provides web and hosting solutions based mainly on *BSD systems. He can be reached at http://www.snb.it.

SnB Qualità Informatica

Web solutions for your business

SnB - Viale Don Minzoni 5/B - Jesi (AN) - Italy
Tel: +39 0731204358 - http://www.snb.it - info@snb.it

What's New in PC-BSD 9.1

PC-BSD 9.1 is expected to be released during September, 2012. This article introduces some of the new features of this release.

PC-BSD 9.1 adds many new features, ranging from more graphical utilities available within Control Panel, a redesigned installer, a server installation wizard, and improved jail management. This article introduces these new features.

New Control Panel Utilities

Control Panel was introduced in PC-BSD 9.0, providing common access to graphical configuration utilities, regardless of the desktop one is logged into.

PC-BSD 9.1 adds several more graphical configuration utilities:

1. The About icon, seen in Figure 1, makes it easy to determine the PC-BSD version, the hostname of the system, the versions of the desktops which are installed, and the version of X that is installed.
2. The Active Directory & LDAP utility, seen in Figure 2, allows you to set the client information for connecting to Active Directory or LDAP servers.
3. The EasyPBI utility started out as a FreeBSD port and is used to automate the conversion of a FreeBSD port into a PC-BSD PBI. It is now available through Control Panel. The improved design supports advanced options such as configuring additional ports to build before or after the PBI build, modifying the desktop and menu entries, and adding post-installation scripts. Once the PBI module is complete, it will package the module so that it can be submitted to the PC-BSD PBI build server. A screenshot is seen in Figure 3.

Figure 1. *About Utility*

4. The GDM Configuration utility, seen in Figure 4, can be used to configure a user account for automatic login. It can also be used to configure remote login through XDMCP.
5. The Hardware Compatibility utility, seen in Figure 5, is available both during installation and afterwards using Control Panel. It provides a quick overview of detected hardware devices and indicates whether or not the system's video, Ethernet, wireless, and sound devices are compatible with PC-BSD.
6. The Mount Tray utility is available in both Control Panel and the System Tray. It allows easy access to mounted partitions and USB drives. If you insert a USB drive, a pop-up message will indicate that a new device is available. If you right-click the Mount Tray, as seen in the example in Figure 6, you can choose to mount or automount the device. You can also access the mounted partitions using the desktop's default file manager by clicking "Open Media Directory".

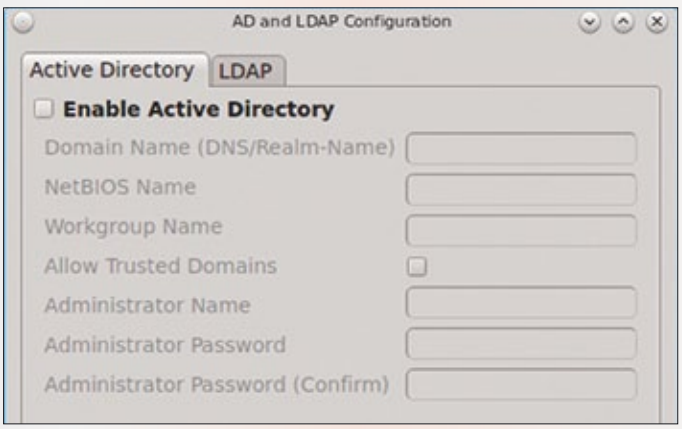


Figure 2. *Active Directory & LDAP Utility*

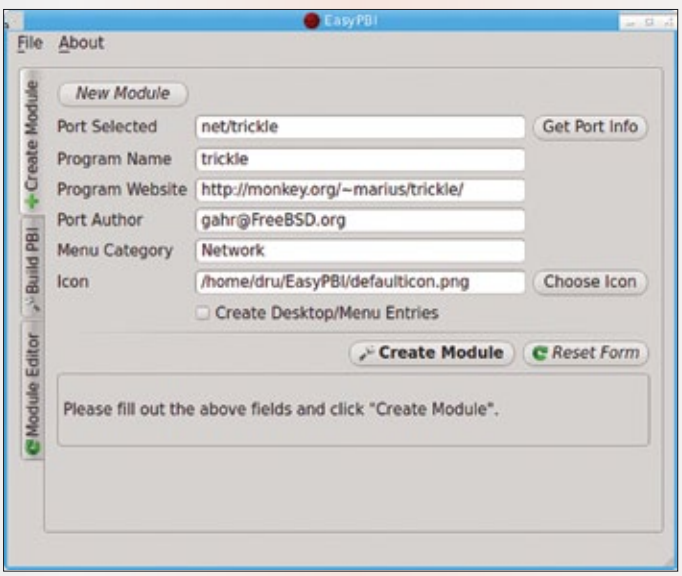


Figure 3. *EasyPBI Utility*

7. The Sound Configuration utility, seen in Figure 7, can be used to test sound or change the default audio device. The drop-down menu can be used to determine which audio devices are available and to change the default device. A "Test sound" button is provided to test the selected audio device.

Redesigned Installer

The PC-BSD 9.1 installer has been redesigned to allow for OEM installs as it separates installation tasks from post-installation configuration tasks. Installation tasks include determining which system components to install and the disk layout to use. Post-installation configuration tasks include setting the timezone, the administrative password, and creating the initial login account. The redesign also simplifies the installation process. A default installation

Figure 4. *GDM Configuration Utility*

Figure 5. *Hardware Compatibility Utility*

can begin after 4 mouse clicks. In PC-BSD 9.1, a default installation is defined as follows:

- installation occurs on the entire primary drive
- if the system has less than 2GB of RAM, that drive is formatted with UFS. Otherwise, it is formatted with ZFS.
- if the system has less than 2GB of RAM, the LXDE desktop will be installed. Otherwise, the KDE desktop will be installed.

For users who wish to change the default installation partition, filesystem, or desktop, each installation screen contains a Customize button. Figure 8 shows the options which are available when the Customize button is selected in the Desktop Selection screen.

The Customize button of the Disk Selection screen of the installer now supports three modes of operation:

- *Basic*: (default) used to specify which partition or disk to install to and to configure encryption.
- *Advanced*: used to specify the installation partition or disk, GPT partitioning, encrypt user data, disable the FreeBSD boot menu, or specify the filesystem to use and the layout of that filesystem.
- *FreeBSD Expert*: used to drop down to a shell to manually enter the commands to configure the disk layout.

ZFS configuration has been improved. If you wish to add multiple drives to the ZFS pool, the installer will indicate the minimum number of drives needed for a mirror, RAIDZ1, RAIDZ2, or RAIDZ3. The installer also allows you to select the following ZFS properties for each ZFS mount point (dataset): atime, canmount, checksum, compression, and exec.

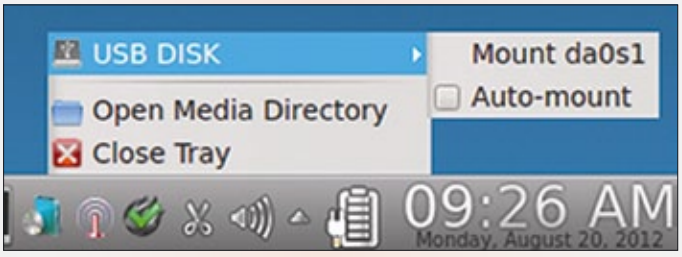


Figure 6. *Mount Tray Utility*

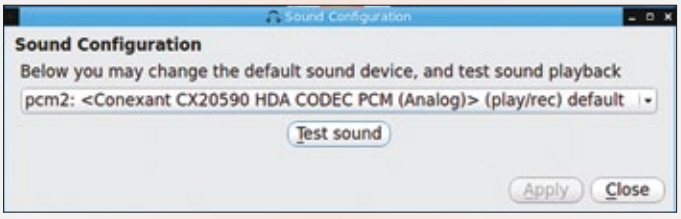


Figure 7. *Sound Configuration Utility*

Server Install Wizard

The redesigned installer also adds a server install wizard capable of installing two types of servers:

- *FreeBSD Server*: installs a basic, vanilla installation of FreeBSD. While the installation routine is different, the end result is the same as if one had installed FreeBSD from a FreeBSD media as it results in a minimal, command line only FreeBSD server installation.
- *TrueOS*™: adds the following features to a vanilla installation of FreeBSD: the PBI Manager command line suite of utilities which can be used to manage PBIs and create one's own software repositories, a command line utility for managing system components, a command line utility for managing updates, and the command line version of Warden® for jail management.

Besides providing a graphical installer, using PC-BSD to install a server offers the following advantages:

- the ability to easily configure ZFS during installation.
- the ability to configure encryption during installation.
- a wizard to configure the server for first use. This wizard is used to configure the system host name, root password, primary login account, enable SSH, configure networking, and install src or ports.

Improved Jail Management

Warden®, PC-BSD's utility to manage jails, has been completely redesigned for 9.1. It no longer needs to be installed as it is part of the base system and available from Control Panel. Some of its new features include the ability to:

Figure 8. *Customizing the Desktop*

- create three types of jails: a traditional FreeBSD jail for running network services, a (less secure) ports jail for safely installing and running FreeBSD ports/packages from a PC-BSD system, and a Linux jail for installing Linux (currently installation scripts are provided for Gentoo and Debian)
- set multiple IPv4 and IPv6 addresses per jail
- quickly install meta-packages of common network server applications on a per-jail basis
- use Update Manager to manage software and system upgrades on a per-jail basis
- use User Manager to manage user accounts on a per-jail basis
- manage ZFS snapshots on a per-jail basis if the PC-BSD system is formatted with the ZFS filesystem
- export a jail which can be then be imported into the same or a different jail

Warden® provides a graphical interface for the PC-BSD desktop and a command line version for a TrueOS™ installation. Figure 9 shows an example of a system with three jails installed, one of each type.

The main screen of Warden® provides an overview of each jail as well as buttons for stopping and starting the highlighted jail.

The tools tab provides the following buttons:

- *User Administrator*: opens User Manager to manage the highlighted jail's user accounts and groups. This button is not available if a Linux jail is highlighted.
- *Service Manager*: opens Service Manager to view which services are running in the jail and to configure which services should start when the jail is started. This button is not available if a Linux jail is highlighted.
- *Launch Terminal*: opens a terminal with the root user logged into the jail. This allows you to adminster the jail from the command line.
- *Check for Updates*: launches Update Manager to determine if any of the jail's installed applications have newer versions available. Update Manager will also indicate if system updates are available to be installed into the jail. This button is not available if a Linux jail is highlighted.
- *Export Jail*: used to save a backup of the jail and all of its software, configuration, and files.

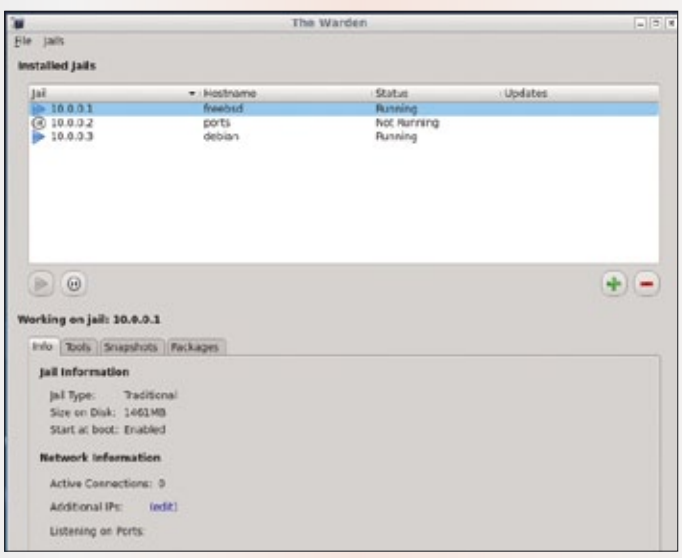


Figure 9. *Warden® Graphical Interace*

If the PC-BSD system was formatted with ZFS, the Snapshots tab can be used to manage snapshots, or point-in-time copies of the filesystem. Since jails share the filesystem used by PC-BSD, any type of jail, including a Linux jail, can take advantage of this ZFS feature. This tab provides buttons to create, delete, restore, mount, and unmount snapshots.

The packages tab allows you to install meta-package software which will be tracked by Update Manager for newer versions. Common server applications are available, such as databases, web servers, file servers, and programming languages.

Summary

PC-BSD 9.1 introduces many new features which are designed to make it easier than ever to install and configure a desktop or server based on FreeBSD. You can learn more about how to use these features in the PC-BSD 9.1 Users Handbook which is provided as an icon on the desktop of an installed release. You can read a preview of this Handbook prior to release at the PC-BSD documentation wiki: *http://wiki.pcbsd.org/index.php/PC-BSD_Users_Handbook*.

DRU LAVIGNE

Dru Lavigne is author of BSD Hacks, The Best of FreeBSD Basics, and The Definitive Guide to PC-BSD. As Director of Community Development for the PC-BSD Project, she leads the documentation team, assists new users, helps to find and fix bugs, and reaches out to the community to discover their needs. She is the former Managing Editor of the Open Source Business Resource, a free monthly publication covering open source and the commercialization of open source assets. She is founder and current Chair of the BSD Certification Group Inc., a non-profit organization with a mission to create the standard for certifying BSD system administrators, and serves on the Board of the FreeBSD Foundation.

Setting up Your

OwnCloud Instance via The Warden™

With the increase of smart-phones, tablets and the general mobility of users, we are also seeing an increase in the interaction with applications and data online – commonly known as the "Cloud".

With this change in behavior, users are becoming increasingly aware of the potential privacy and security issues that are associated with personal data being stored offsite. Companies, governments, or even nefarious individuals could easily obtain access to this data for whatever purposes they so deem. With this ongoing trend, we have begun to see new software become available which allows users to host their own private "Cloud" at whatever location they wish, even from their own home desktop or server. In this article we will be taking a look at the OwnCloud software, specifically how to do the initial installation and configuration inside a jail run by PC-BSD's® jail management utility, the Warden™. First we will take a look at a setup done from a PC-BSD graphical interface, and then explore the same setup from the command-line using TrueOS™, the server version of PC-BSD.

Figure 1. *Starting the Warden GUI*

Figure 2. *Assigning an IP address and hostname*

Creating the jail via the Warden GUI

To begin, you will need to start the Warden GUI via the Control Panel (Figure 1) Next, you will want to ensure that the jail is running on the correct network interface in Jails → Configuration. Via the pull-down menu, select the network interface you want to run your jails on. Normally the selected interface should be the same interface you are using to connect to your network and the Internet.

Once the jail interface has been set properly, go to File → New Jail to start the creation process. On the first screen, you will need to assign an IP address and hostname to this new jail, and then click "Next" to continue (Figure 2).

Note

The IP address should be a unique address on your network, not the same as your host's IP. For example, if your system's IP address is 192.168.0.100, then you could pick something unused in the 192.168.0.xxx range, or another address on the same subnet as the host. Next, select the type of jail you are creating. In this instance, you will be using a "Traditional Jail", select it and click "Next" to continue (Figure 3). On the next screen, you will need to enter a root password for this jail and click "Next" to continue.

Lastly you can set additional options for this jail. If you plan on building software from FreeBSD ports, you can select to install the ports tree, and system sources. If you want this jail to run every time the system boots, you may wish to also check "Start jail at system bootup". When you are ready, click "Finish" to begin the jail creation (Figure 4). This may take a few minutes the first time you create a jail, because a fresh jail environment needs to be downloaded.

After the jail creation has finished, you will then need to install the software required to run OwnCloud. OwnCloud is written in PHP, and requires access to a database, such as MySQL. In addition you will need a web-server, such as Apache, to serve the site.

Using the Warden GUI, you can select your new jail and click "Packages" to browse for and select the packages MySQL, PHP and Apache to the jail. Click "Apply" to begin installing them (Figure 5). Once the packages have finished installing, start the Apache and MySQL services inside the jail. You can do so on the "Tools" tab of the jail manager by selecting the "Service Manager" button (Figure 6).

Figure 3. *Jail type selection*

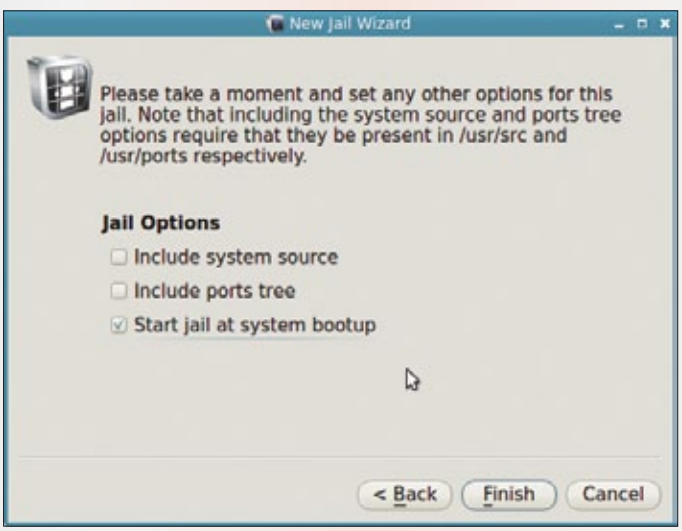


Figure 4. *Setting the jail options*

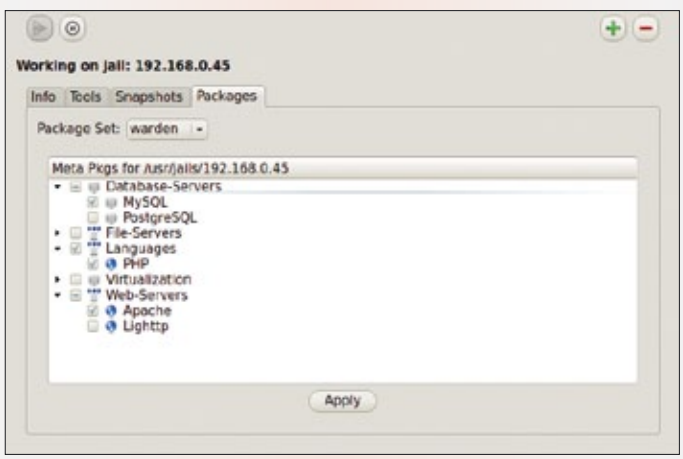


Figure 5. *Selecting the server packages required for OwnCloud*

Figure 6. *The available tools for a FreeBSD traditional jail*

In the Services Manager, you will be given a list of services for this jail. Scroll through the list, select the "apache22" service, and click "Enable" and then "Start" (Figure 7). Repeat the process for the mysql service.

If everything has been successful to this point, you should be able to bring up your web-browser, point it to the jail's IP address, and see the text "It works!" Congratulations, you are now ready to setup your OwnCloud software.

Note

If you are trying to connect from a web-browser on another system, you may need to open port 80 in the Control Panel → Firewall utility first). You may skip the next section and jump down to *Configuring the jail for OwnCloud*.

Creating the jail via the Warden Command-Line

Users who do not wish to run a full desktop operating system may still use the Warden via a command-line interface after installing TrueOS (which is included on the PC-BSD install DVD). To begin, you will need to configure it to use the correct network interface for your jails. This is done by editing the file `/usr/local/etc/warden.conf`, and changing the interface line as shown below:

```
NIC: re0
```

With this configured, you are now ready to create the new jail. Use the following command, changing the hostname / IP address to your preference.

```
# warden create 192.168.0.45 owncloudjail --startauto
```

With the jail now created, you will need to install the packages required for running an OwnCloud Server. Using the built-in `pc-metapkgmanager` command, you can do so with the following command:

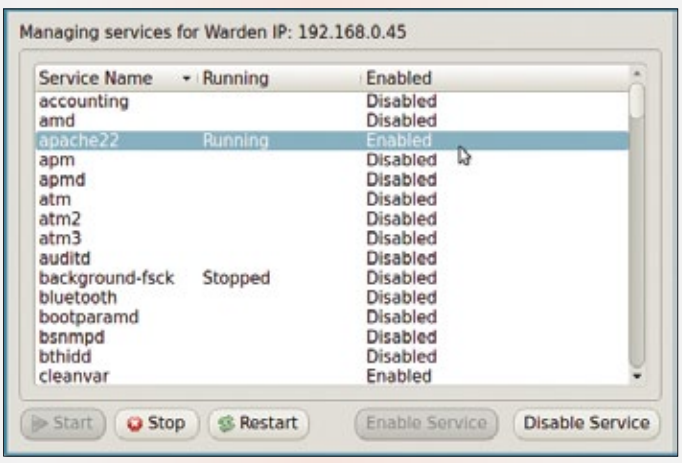


Figure 7. *Enabling the services for this jail*

```
# pc-metapkgmanager --pkgset warden --chroot
/usr/jails/192.168.0.45 add MySQL,Apache,PHP
```

Once the packages have finished installing, you will need to enable them to run at startup by editing /etc/rc.conf from within the jail. This can be done using the following commands and by adding the lines `apache22_enable="YES"` and `mysql_enable="YES"` to `/etc/rc.conf`.

```
# warden chroot 192.168.0.45
root@owncloudjail:/ # vi /etc/rc.conf
root@owncloudjail:/ # /usr/local/etc/rc.d/apache22 start
root@owncloudjail:/ # /usr/local/etc/rc.d/mysql-server start
```

Congratulations, you are now ready to setup your OwnCloud software.

Note

If you are trying to connect from a web-browser on another system, you may need to open port 80 in the Control Panel → Firewall utility first). If running from TrueOS™ you may need to add an exception into `/etc/pf.conf`.

Configuring the Jail for OwnCloud

To install the OwnCloud software, fetch it and extract it into the jail via the shell prompt. To open a shell, navigate back to the "Tools" tab of the jail and click "Launch Terminal", or from the command prompt run `warden chroot 192.168.0.45` replacing with your IP. Once the shell or terminal has started, type the following commands to download and extract your OwnCloud.

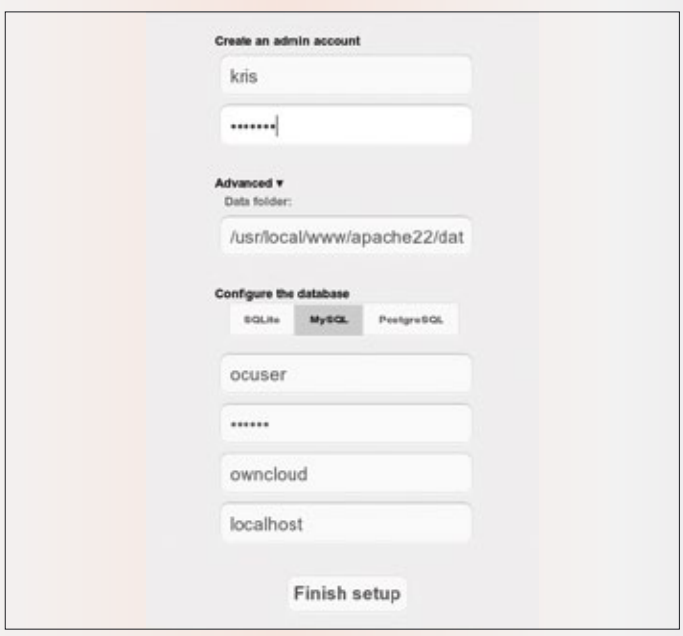


Figure 8. *OwnCloud Setup Screen*

```
# cd /usr/local/www/apache22/data
# fetch http://download.owncloud.org/releases/owncloud-
                    4.0.6.tar.bz2
# tar xvf owncloud-4.0.6.tar.bz2
# chown -R www:www owncloud
```

With this done, you will now have a new `/usr/local/www/apache22/data/owncloud` directory, waiting to be setup. Before you close the shell, create a new MySQL database, and configure PHP properly for your OwnCloud instance. To create the database, use the commands below, changing the password to one of your liking.

```
# mysql -u root
mysql> create database owncloud;
mysql> grant all on owncloud.* to ocuser@localhost
                    identified by "mypass";
mysql> quit
```

After configuring MySQL, you need to enable some additional Apache PHP options. Open the file `/usr/local/etc/apache22/httpd.conf`, using your favorite editor, such as "vi" or "edit", and browse for the following section:

```
# AddType allows you to add to or override the MIME configuration
# file specified in TypesConfig for specific file types.
#
#AddType application/x-gzip .tgz
#
```

Add the following lines right below this section:

```
# AddType allows you to add to or override the MIME configuration
# file specified in TypesConfig for specific file types.
#
#AddType application/x-gzip .tgz
#
AddType application/x-httpd-php .php
AddType application/x-httpd-php-source .phps
```

Lastly, search for the following section:

```
<IfModule dir_module>
    DirectoryIndex index.html
</IfModule>
```

and add:

```
<IfModule dir_module>
    DirectoryIndex index.html index.php
</IfModule>
```

With these changes made, save the httpd.conf file, then restart Apache with the following command:

```
# /usr/local/etc/rc.d/apache22 restart
```

With this configuration done, you are now ready to launch OwnCloud. In your browser, navigate to the URL: *http://192.168.0.45/owncloud/*, replacing "192.168.0.45" with your jail IP. When you bring up the page, you will be presented with a first-time setup screen. Create a new user and password, and be sure to click the advanced button. In the advanced settings you will need to enter the MySQL username, password, and database name you previously created (Figure 8). Click "Finish" to finalize the OwnCloud configuration, and enter your new cloud interface!

With OwnCloud setup and configured properly, you should be taken to the main interface screen (Figure 9). From here you can now begin to use it to store files (ala DropBox), manage your calendars, contacts, and much more.

By clicking the small "gear" icon in the bottom left, you can further customize your Cloud account, locate the CalDav, CardDav and WebDav addresses for mobile devices, install 3rd party applications and more. For more information on using the OwnCloud interface and integrating with your mobile device, you may wish to read through the documentation and guides located on the OwnCloud support site. (*http://owncloud.org/support/*).

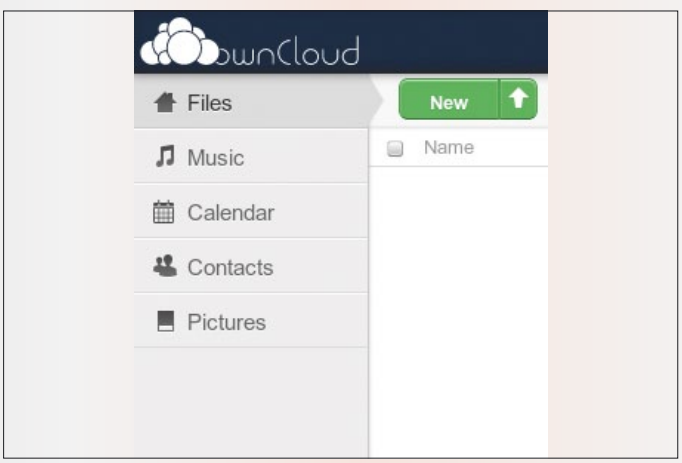


Figure 9. *OwnCloud main screen*

KRIS MOORE

Kris Moore is the founder and lead developer of PC-BSD. He lives with his wife and four children in East Tennessee (USA), and enjoys building custom PC's and gaming in his (limited) spare time. kris@pcbsd.org

Unix IPC with Pipes

This article explains one of the earliest forms of inter-process communication (IPC) in Unix. Pipes were the original form of Unix IPC and were present in Third Edition of Unix (1973). They can only be used to communicate between related processes, but despite this limitation they still remain one of the most frequently employed mechanisms for IPC.

What you will learn…

- How pipes are created
- How processes use pipes
- File descriptors
- The 'fstat(1) command

What you should know…

- Basic command line operations

The `fstat(1)` command, which first appeared in 4.3BSD, displays the status of open files, sockets and pipes (as well as other objects) on a system and provides information on their I/O activity. To understand the output of `fstat(1)` it is necessary to know what a 'descriptor' is and how it is used to identify an access path for I/O from a userland program to a disk, network socket, pipe, etc.

Descriptors

Descriptors are used within programs to reference 'objects' used for I/O. Typically, these objects refer to files, pipes or sockets; less common are event queues for notification of kernel events, and the 'crypto' object which is used for direct access to cryptographic hardware. Additional types exist, but their definition and presence varies from one BSD to the other.

The descriptor is an integer which is allocated by the kernel when a program executes the appropriate system call to open the the object: `pipe()` for opening a pipe, `open()` for opening a file, or `socket()` for local or network sockets, etc. All system calls which perform I/O on the given object or modify its parameters will reference the object using the descriptor. A descriptor remains allocated ('open') until it is either closed by the process or the process exits.

Most applications, including shells, associate file descriptors 0,1,2 with standard input, standard output, and standard error, respectively. There is a limit on how many descriptors a process may have open at any given time. This is defined by the OPEN_MAX constant, and ranges from 64 (FreeBSD) to 128 (NetBSD) and on OpenBSD, from soft limit of 64 to a hard limit of 1024.

Associated with every process is a table of open descriptors, and each descriptor is merely an index into this table. The descriptor table holds a reference to a *file entry*. The kernel maintains a table of *file entries* for all open objects in the system. The *file entry* itself is an instance of the *file structure*. A field in the *file structure* identifies the type of underlying object – *socket* or *pipe* for sockets or pipes respectively, type *v-node* for files in the file system, which may include FIFOs and devices in `/dev/`, etc. (The possible values for this field vary among BSDs; it is necessary look in sys/file.h on the particular system to see how they are defined.)

The *file structure* also holds the *status flags* for the object (e.g., read only, read-write, append, etc) specified when the object was opened, the current offset within the file where the next read or write will occur (if the object referenced is a file), the amount of data transferred and the number of transfers, and the particular I/O routines specific to that type of object.

It is important to note that when a process calls `fork(2)`, the open descriptors in the parent process are copied to the child, and after the `fork()` the parent and child will

share the same descriptors for reading and writing data. This will also be the case if the child calls `exec(3)` to execute a program different from that of the parent, though this behavior can be changed by setting the 'close-on-exec' flag which is associated with a descriptor. If the flag is set, open descriptors inherited from the parent will be closed when calling `exec()`.

After a `fork()`, both the descriptors in the parent and the child will reference the same *file entry*. This means that reads or writes by either process will advance the offset where the other process will perform its next read or write. Also, the values for I/O activity will be incremented by either process. The fstat(1) program is a tool for reading the table of open descriptors for a given process and returning statistics on their I/O.

The example in Listing 1 uses fstat(1) on OpenBSD. In another terminal, the pager program `/usr/bin/less` is running and its PID is passed to fstat as an argument. The options are:

s – report file I/O statistics – the number of transfers and number of kilobytes transfered. This option produces no output unless fstat is run as the super-user, or the UID of the process is the same as the UID of the user running fstat.
o – report file offset. This is the byte offset from the beginning of the file where the process is either reading or writing.
p – the pid of the process

The column headings in the output are:

USER – the owner of the process
CMD – the command
PID – the process ID
FD – the file descriptor number, or one of the following special names:
text – executable text inode
wd – current working directory
root – root inode
tr – kernel trace file (the output file if ktrace is running)
MOUNT – mount point for file system where the particular file resides
INUM – inode number for the particular file
MODE – file type and permissions on the file
R/W – whether file is open for reading and/or writing
SZ/DV:OFFSET – if a regular file, this will be the size of the file followed by the current offset into the file where the next read or write will occur; if a character or block special file, the name of the device file in /dev
XFERS – the number of times data has been transferred in either direction.
KBYTES – number of kilobytes transferred.

N.B. Due to a small bug OpenBSD 5.1, the value for KBYTES are incorrect. This will be fixed in the new 5.2 release due in November. Until the release of 5.2, the two required patches can be downloaded from *www.tetrardus.net/bsdmag/diffs/*.

The first two lines of Figure 1 show information about the binary executable (`/usr/bin/less`) and its working directory. Most of the information in these two lines can be obtained using 'ls -li' on `/usr/bin/less` or the current working directory.

The following 3 lines show information for file descriptors (FD) 0, 1, 2, which correspond to standard input, standard out and standard error. These 3 descriptors map to the same inode, 964 (INUM), which refers to a file of type *character* ('c' in MODE column) and is the device file `/dev/ttyp9` associated with the terminal where 'less' is running. Since 'less' was created via the `fork()` and `exec()` method initiated by the shell running in the other terminal and con-

Listing 1. *Output of /usr/bin/fstat*

```
# fstat -sop 'pgrep less'

USER    CMD      PID     FD  MOUNT    INUM  MODE        R/W   SZ|DV:OFFSET    XFERS      KBYTES
paul    less    8348   text  /usr   978510  -r-xr-xr-x    r   13364:0             0           0
paul    less    8348     wd  /home  987392  drwxr-xr-x    r     512:0             0           0
paul    less    8348      0  /         964  crw--w----   rw    ttyp9            363          19
paul    less    8348      1  /         964  crw--w----   rw    ttyp9            363          19
paul    less    8348      2  /         964  crw--w----   rw    ttyp9            363          19
paul    less    8348      3  /        1616  crw-rw-rw-    r      tty              0           0
paul    less    8348      4  /var       12  -rw-r--r--    r    25708:8192         2           8
```

sequently inherited the shell's descriptors that were attached to the device ttyp9, both processes now share the same *file entry* for this device. Therefore the values under XFERS and KBYTES reflect not only I/O activity initiated by 'less' but also the I/O activity generated by the shell, including activity that occurred before 'less' was started.

The next line shows that the program 'less' has opened `/dev/tty` (INUM = 1616) for reading (R/W = r) on descriptor 3. It is through this descriptor that 'less' reads input from the keyboard. Thus far, there has been no input from the keyboard (XFERS = 0, KBYTES= 0).

The last line shows that 'less' has a file open on descriptor 4. The file's inode number is 12, the file is open for reading, the file is 25708 bytes in size, and the file's current offset is 8192 bytes into the file (SZ|DV: OFFSET = 25708:8192). 8kb has been read (KBYTES = 8) and this required 2 data transfers.

Again, much of this information is static (e.g., file's inode, size, etc) and can be obtained using options to `/bin/ls`.

Pipes

Pipes provide a fast, reliable, stream oriented method of uni-directional data flow between *related* processes. In this case, *related* specifically means processes having a parent-child relationship or processes having a sibling relationship (i.e processes that have a common ancestor) (Figure 1).

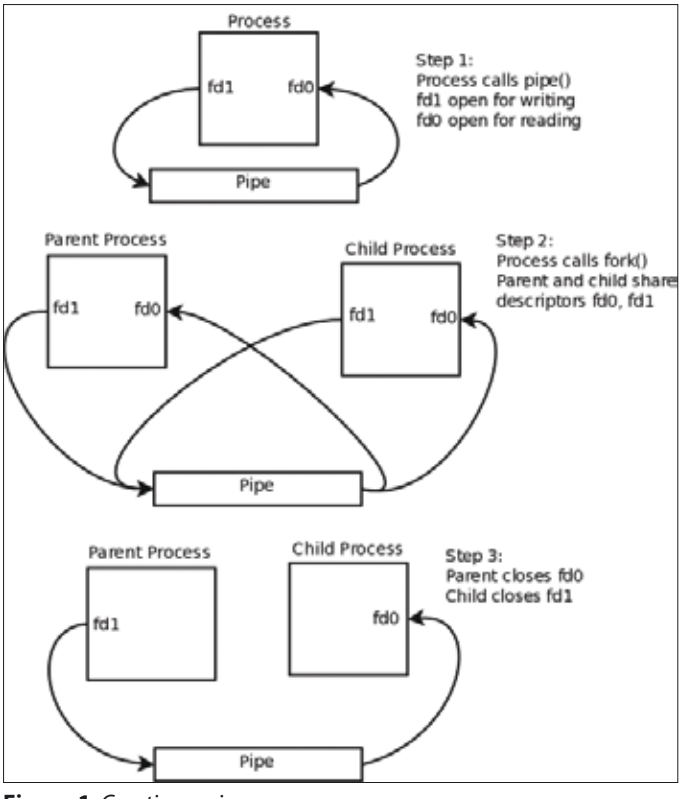


Figure 1. *Creating a pipe*

Pipe creation takes advantage of the fact that, after a call to `fork()`, open descriptors in a parent process will be inherited by the child. Figure 1 shows the three step process that creates a pipe. First, a process invokes the `pipe(2)` system call, which creates a buffer in the kernel and returns 2 file descriptors which reference the pipe. The descriptor `fd0` is open for reading, and `fd1` is open for writing. Data written to descriptor `fd1` can be read on `fd0` (the arrows indicate the direction of data flow).

In step 2, the process calls `fork()`, and the child inherits the parent's open descriptors. After the fork, both the parent and the child can read or write to the pipe.

In the final step, the parent then closes the descriptor which is open for reading (`fd0`) and the child closes the descriptor which is open for writing (`fd1`). The result is a uni-directional data path between fd1 in the parent to `fd0` in the child. If the child were to `exec()` another program, the pipe would still be open, and the new process would read its standard input from the pipe.

A pipe is a buffer in kernel memory which defaults 16kb in size. This buffer exists until both descriptors are closed. Data written into the pipe is stored in this buffer until it is read by the process on the other end. If the processes on both ends close the descriptors associated with the pipe, then any data left in the buffer is discarded.

If only one end of the pipe has been closed, the pipe is 'widowed'. A process writing to a pipe after the read-end has been closed will receive a SIGPIPE signal from the kernel. The default action for this signal is to terminate the process. A process reading from a pipe whose write-end has been closed will read any remaining data in the pipe buffer after which it will receive an EOF (end of file). The pipe is then in a 'end-of-file' state and will remain in this state until the last descriptor is closed.

Since the data flow between two process occurs within the kernel on the same host, the data transfer is reliable and data cannot be lost. It is also stream oriented, that is, the process reading data from the pipe cannot determine any boundaries in the data based upon writes performed by the other process.

Because open descriptors are copied across a call to `fork()`, it is possible to have multiple processes reading from or writing to either end of a pipe. The common case is one where a pipe has multiple writers and only a single reader. For instance, a typical configuration of Apache http server will spawn several children to serve client requests concurrently, and each child will write its log data to a single instance of a log processing program (e.g., cronolog). This is illustrated in Figure 2.

The upper part shows the relationships after the httpd process has set up the pipe, but before it has spawned

any children. The preliminary steps are identical to what was shown in Figure 2: the httpd process called `pipe()` to create the two descriptors; this was followed by a call `fork()` to create a child process. The child and parent then closed the appropriate descriptors to create the uni-directional pipe. Then the child called `exec()` to replace itself with the cronolog program.

The lower part of Figure 2 shows the relationships after the httpd daemon has forked 4 children. Each child has inherited the open descriptor connected to the write-end of the pipe and cronolog is able to read the output from any of the httpd processes which send data to the pipe.

An important property of pipes is that of atomicity. If more than one process is writing to a pipe, then there must be some protection against data from two separate processes being interleaved in the pipe buffer. To mitigate this problem, the kernel guarantees the atomicity of writes which are less than a predefined size as set by the PIPE_BUF constant. On OpenBSD, NetBSD, and FreeBSD PIPE_BUFF is set to 512 bytes. (On linux, this value is much larger: 4096 bytes). This means that if a process writes data which is less than or equal to PIPE_BUF bytes, then either all of the data will be written or none of it will. This prevents data from two processes being intermixed within the pipe buffer, as one process will write all of its data first before the second is allowed to write anything. Another consequence of atomicity is that if the amount of data to write is less than or equal to PIPE_BUF, but larger than the amount of free space in the buffer, then the write will not occur until there is enough space in the buffer to perform the write atomically.

Most sysadmins are familiar with pipes through their use on the command line to redirect the standard output of one program to the standard input of another program (e.g., cat `<file> | grep <something>`). In this case, the shell executing the commands uses a series of `fork()`s and `exec()`s along with the `pipe()` system call to set up the pipes so that the standard output (descriptor 1) of the first command writes to the pipe and the standard input (descriptor 0) of the second command reads from the pipe.

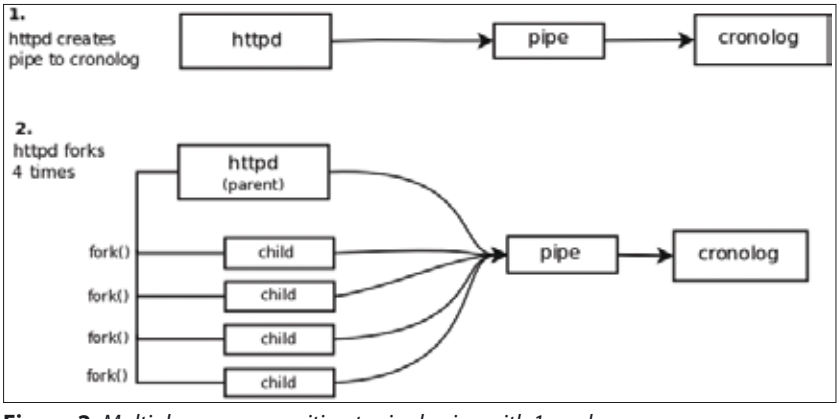


Figure 2. *Multiple processes writing to single pipe with 1 reader*

Listing 2 is the output of fstat for exactly this scenario. In another terminal (not shown), the commands 'cat file_1 | less' are executed and in a second terminal, fstat is invoked with the PIDs of the running 'cat' and 'less' processes as arguments. The upper part of Listing 2 shows the output of 'fstat' after the 'cat' program has read data from the file on disk and written it to the pipe, and 'less' has read from the pipe and filled the terminal with the first portion of the file (The first thing to note is that the column headings in the output don't always apply to the lines containing descriptors which refer to pipes. Also, in this example the output of fstat has been piped to grep to remove lines which are not germane to the discussion).

The uppermost output shows the open descriptors for the 'cat' program. Here, we're interested in descriptors 1 and 3. Descriptor 3 is reading from a file whose inode is 337826 and is located in a directory somewhere in the file system mounted on `/home`. The columns SZ|DV:OFFSET show that the file is 5131637 bytes in length, and the current offset into the file is 32768 bytes (32kb). The XFERS and KBYTES columns show that there have been 2 data transfers from the file on disk which has resulted in 32kb having been read. This corresponds with the current offset in the file, which is also 32kb.

Descriptor 1 (FD 1), standard output, is writing to a pipe. Pipes are uniquely identified by a hexadecimal value. To the right, the values for XFERS and KBYTES are 1 and 16. There has been 1 write operation of 16kb to the pipe.

The second invocation of fstat is on the 'less' program. File descriptor 0, standard input, is reading from the pipe. Here we see the value for 'state:' which is 'W', meaning a write operation is blocked waiting for the reader to read more data from the pipe. Again, the XFERS and KBYTES values show that the 'less' program has read 8kb of data from the pipe. Since the device ttyp7 on descriptors 1 and 2 is also being used by the shell, these XFERS and KBYTES values reflect prior I/O activity *and* activity generated by 'less'.

Taken together, the output shows that 'cat' first read 32kb from the file into its internal buffers and then wrote 16kb to the pipe. The size of the pipe buffer is 16kb so 'cat' filled the pipe to its maximum capacity. 'less' read only the first 8kb of data from the pipe, leaving 8kb in the pipe. We know that 'less' has written data to 'standard output' on descriptor 1 (because we can see the contents of the file in the terminal window), but we don't know exactly how much was written because the XFERS and

Listing 2. *Pipe I/O statistics output by fstat*

```
# fstat output after command 'cat file_1 | less'

fstat -sop 'pgrep cat' | grep -ve text -ve wd

USER   CMD      PID FD  MOUNT    INUM    MODE        R/W   SZ|DV:OFFSET    XFERS      KBYTES
paul   cat    14875      0  /         960    crw--w----    rw    ttyp7            44           6
paul   cat    14875      1  pipe 0xffffffe80b1f37da0 state:                      1          16
paul   cat    14875      2  /         960    crw--w----    rw    ttyp7            44           6
paul   cat    14875      3  /home  337826    -rw-rw-rw-    r    5131637:32768     2          32

fstat -sop 'pgrep cat' | grep -ve text -ve wd

USER   CMD      PID     FD  MOUNT    INUM  MODE        R/W   SZ|DV:OFFSET    XFERS      KBYTES
paul   less    11905     0  pipe 0xffffffe80b1f37da0 state:  W                   1           8
paul   less    11905     1  /         960  crw--w----    rw    ttyp7            44           6
paul   less    11905     2  /         960  crw--w----    rw    ttyp7            44           6
paul   less    11905     3  /        1616  crw-rw-rw-    r      tty             0           0

# fstat output after paging farther into file

fstat -sop 'pgrep cat' | grep -ve text -ve wd

USER   CMD      PID     FD  MOUNT    INUM    MODE        R/W   SZ|DV:OFFSET    XFERS      KBYTES
paul   cat    14875      0  /         960    crw--w----    rw    ttyp7          1259         925
paul   cat    14875      1  pipe 0xffffffe80b1f37da0 state:                     59         944
paul   cat    14875      2  /         960    crw--w----    rw    ttyp7          1259         925
paul   cat    14875      3  /home  337826    -rw-rw-rw-    r    5131637:983040    60         960

fstat -sop 'pgrep cat' | grep -ve text -ve wd

USER   CMD      PID     FD  MOUNT    INUM  MODE        R/W   SZ|DV:OFFSET    XFERS      KBYTES
paul   less    11905     0  pipe 0xffffffe80b1f37da0 state:  W                 116         928

paul   less    11905     2  /         960  crw--w----    rw    ttyp7          1259         925
paul   less    11905     3  /        1616  crw-rw-rw-    r      tty           151           0

#fstat output after paging to end of file

fstat -sop 'pgrep cat' | grep -ve text -ve wd

USER   CMD      PID     FD  MOUNT    INUM  MODE        R/W   SZ|DV:OFFSET    XFERS      KBYTES
paul   less    11905     0  pipe 0xffffffe80b1f37da0 state:  E                 628        5011
paul   less    11905     1  /         960  crw--w----    rw    ttyp7          6671        5018
paul   less    11905     2  /         960  crw--w----    rw    ttyp7          6671        5018
paul   less    11905     3  /        1616  crw-rw-rw-    r      tty           835           0
```

KBYTES values include prior I/O activity generate by the shell. However, since we know that 'less' read 8kb from the pipe, and the value for KBYTES on descriptor 1 is only 6, we can conclude that 'less' didn't write all the data it had in its internal buffers.

The second part of Listing 2 shows the 'fstat' output for the same processes after paging substantially farther down into the file. 'cat' has read 960 bytes of data from the file on disk on descriptor 3, the new offset is at byte 983040, and 940kb of data has been written to the pipe on descriptor 1. 'less' has also read 928kb from the pipe and written that to the terminal.

The lower portion of Listing 2 shows 'fstat' output after paging to the very end of the file. The 'cat' program, after writing the last amount of data to the pipe, exited and the pipe is now 'widowed' (the pipe's state is 'E'). The 'less' program performed a total of 628 read operations on the pipe, transferring 5011kb of data.

Although the illustration of pipes in this article has been limited to half-duplex communication between processes, it is possible, to establish full-duplex inter-process communication using two pipes, one for each direction of data flow.

Summary

Pipes are the most basic type of Unix IPC, and one of the most commonly used mechanisms for passing data between programs. They offer fast, reliable data transfer between related processes. Within a program, a pipe is referenced using a file descriptor. File descriptors identify instances of objects which are used for I/O in a program. The fstat program is a tool that reads the table of open descriptors and returns information about the objects they reference.

PAUL MCMATH

Paul McMath has worked as a Unix admin for 10+ years in Europe and the United States. He has been using one BSD variant or another as his OS of choice since 2002.

FreeBSD

Enterprise Search with Apache Solr Part 1

Back office integration and cross platform search has always posed major challenges especially in large organizations with many legacy systems. With Apache Solr these barriers can be overcome and the power of enterprise search realised.

What you will learn…

- How to commission an Apache Solr search engine

What you should know…

- BSD administration skills

Glueware, middleware – call it what you want – is the bread and butter of the well connected enterprise. Legacy systems, which may not have the benefit of open API's, vendor support or even an import / export facility challenge the systems integrator with a major paradox. Often these systems are critical to the business, but are so culturally embedded in the business model that to replace them is unthinkable, either on the basis of functionality (The users like it) or cost (Too expensive to replace). Worst still, an organisation

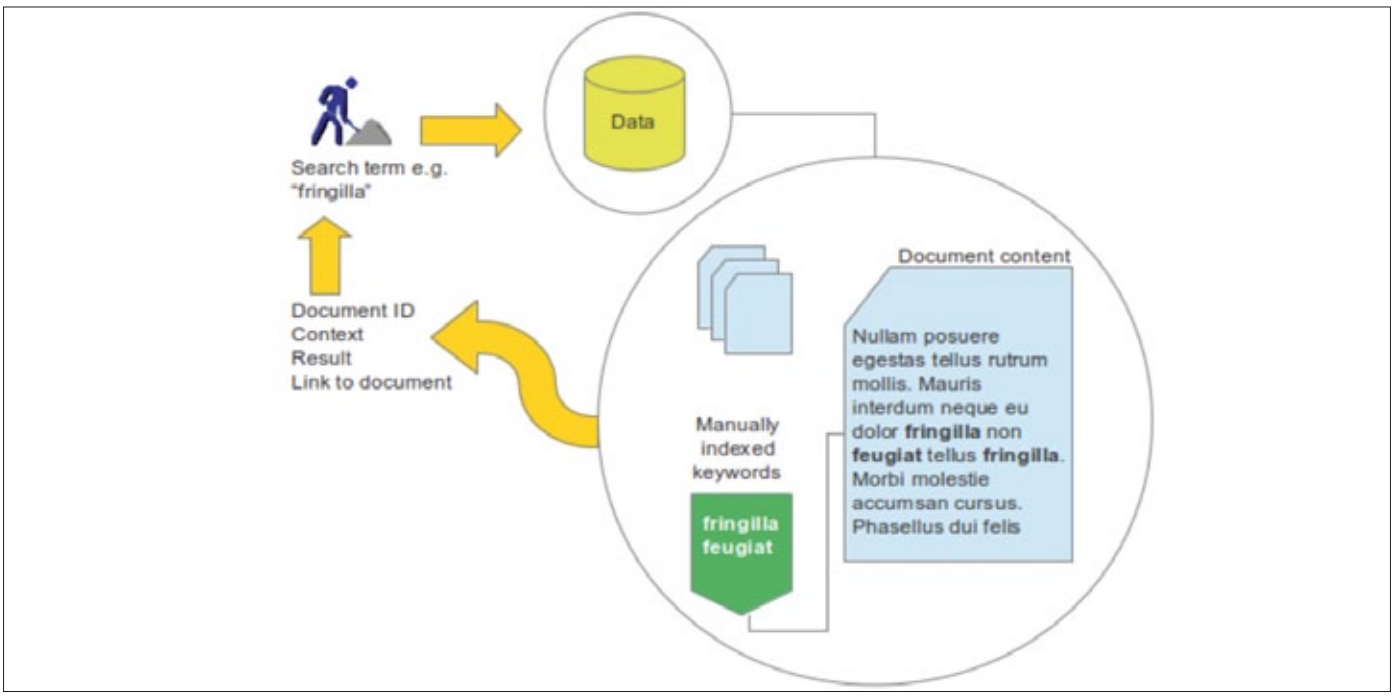


Figure 1. *Traditional search using manually indexed terms*

can reach a dead-end in that the system is no longer maintainable or extensible but at the same time extra functionality is essential – e.g. exposing back office data to web.

Traditionally, integration is accomplished by batch operations, import / export of data, using a messaging system or some form of trigger e.g. a request for specific data via XML. This is fine if the data set is well defined and we are working with "known knowns" and this model works well for integration as well as search. For example, searching for a known surname in a surname field the user searches for a surname "Somerville", and will expect either a match, multiple matches or no result. Unless there is some other search technology applied the user will not quickly find "Sommerville", "Somervile" or even "Summerville". If we take a step back from the historical methods (Figure 1), it is clear that a new search paradigm is now being adopted by innovators on the web – faceted and intelligent ("deep") search. This new search is a mixture of technologies (e.g. Ajax, XML), data structure (Pre-defined, undefined), indexing (Static, dynamic) which revolutionizes the way the user engages with the search process itself. No longer is

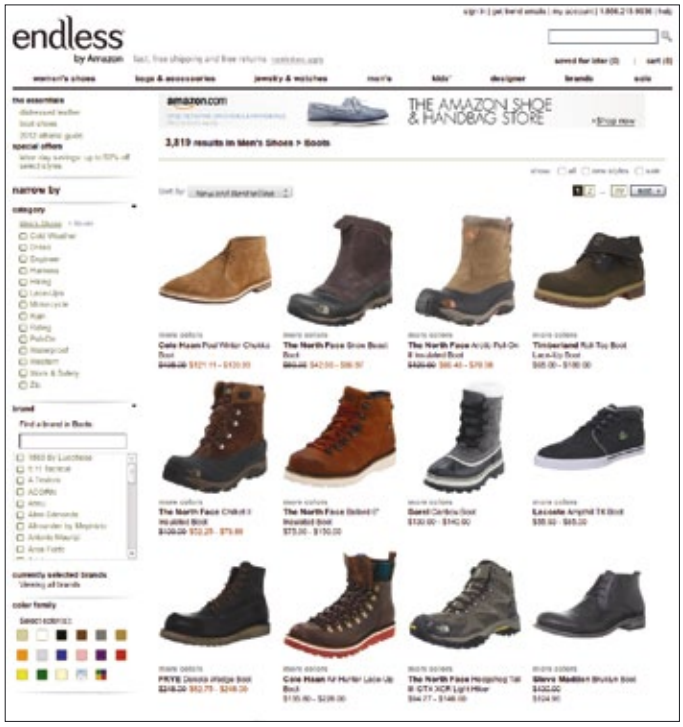


Figure 3. *Example of an innovative website with faceted search*

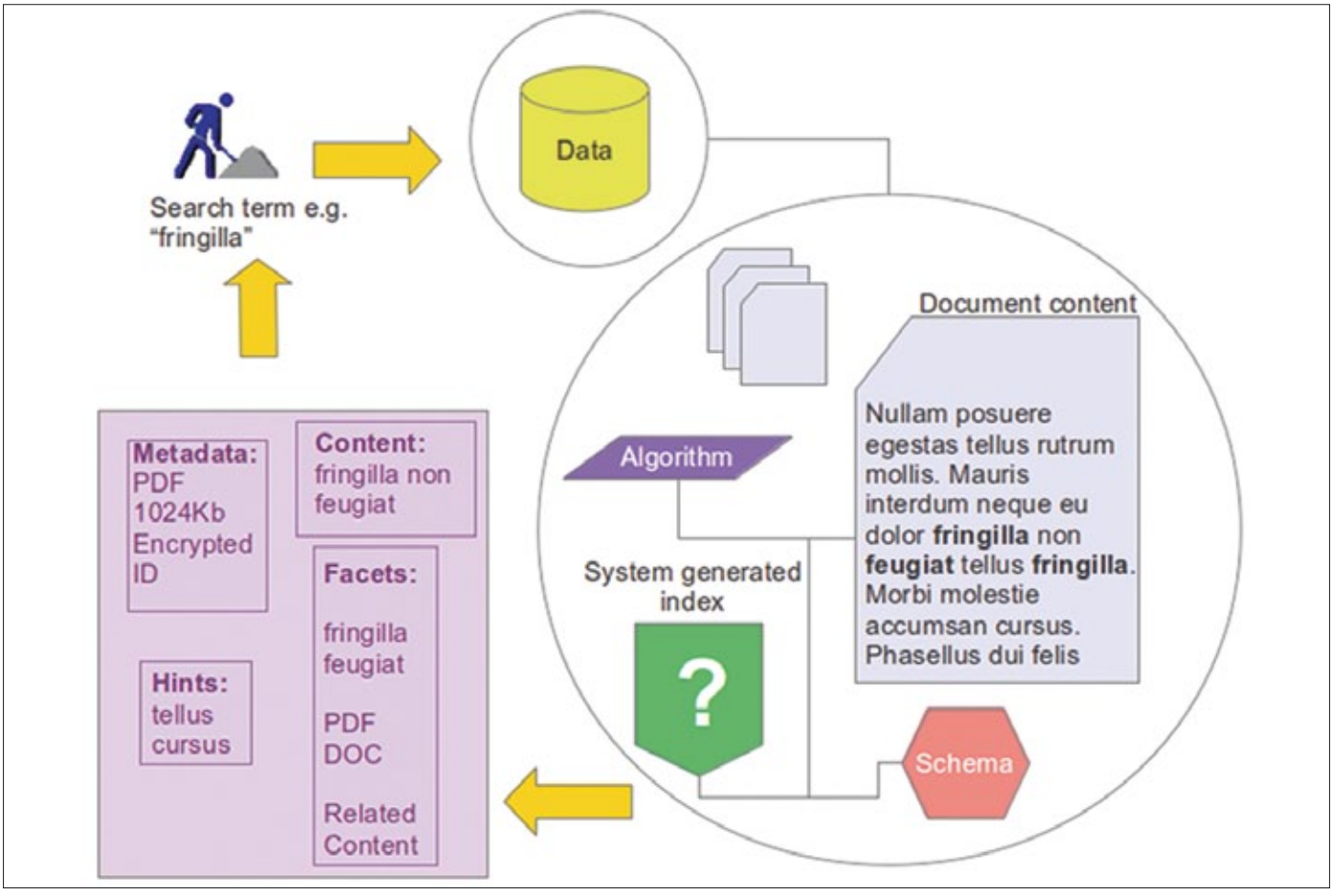


Figure 2. *Solr faceted search, algorithms and schema's. The format of the binary index is dictated by Apache Lucene*

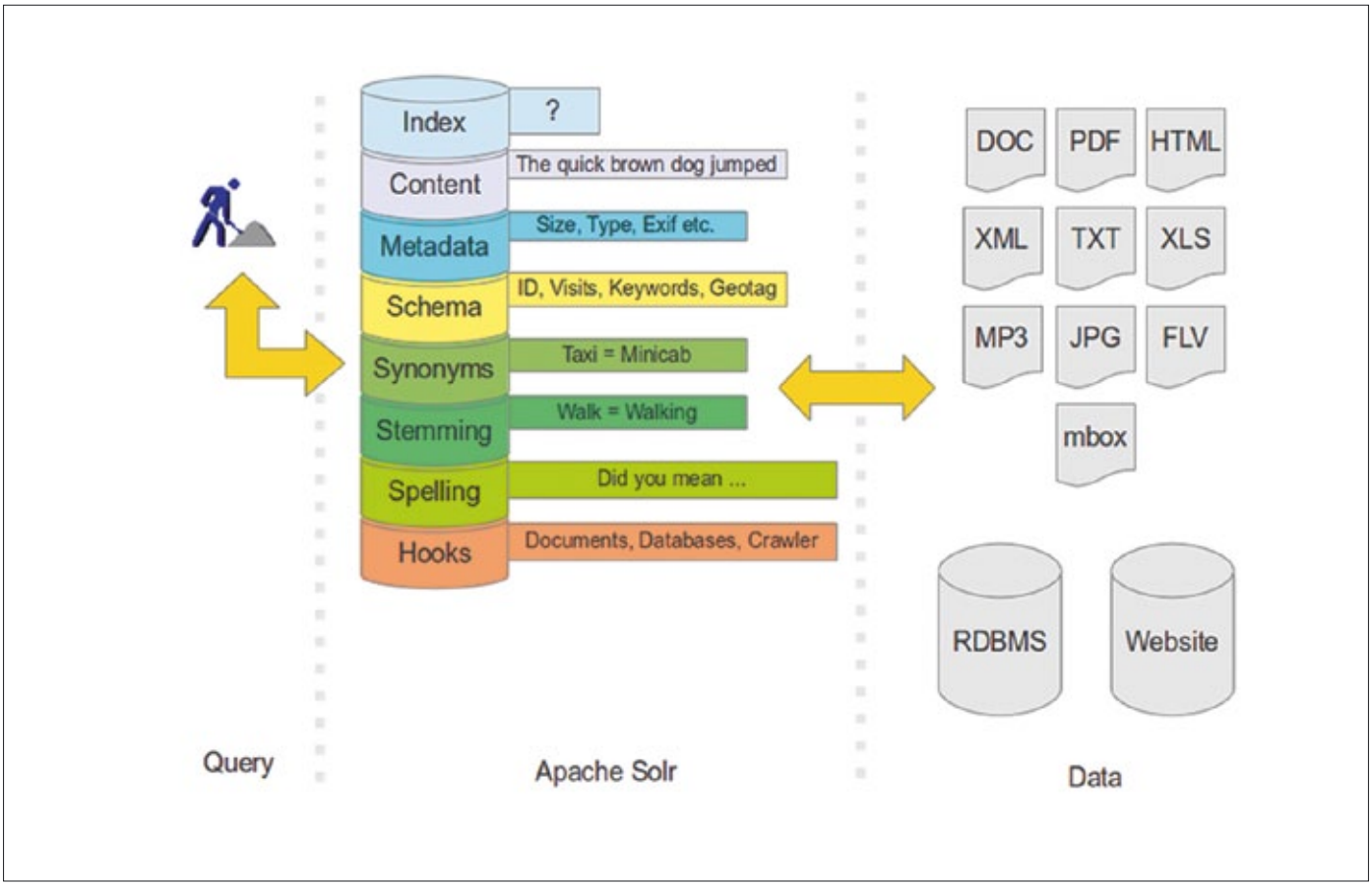


Figure 4. *Enterprise search example*

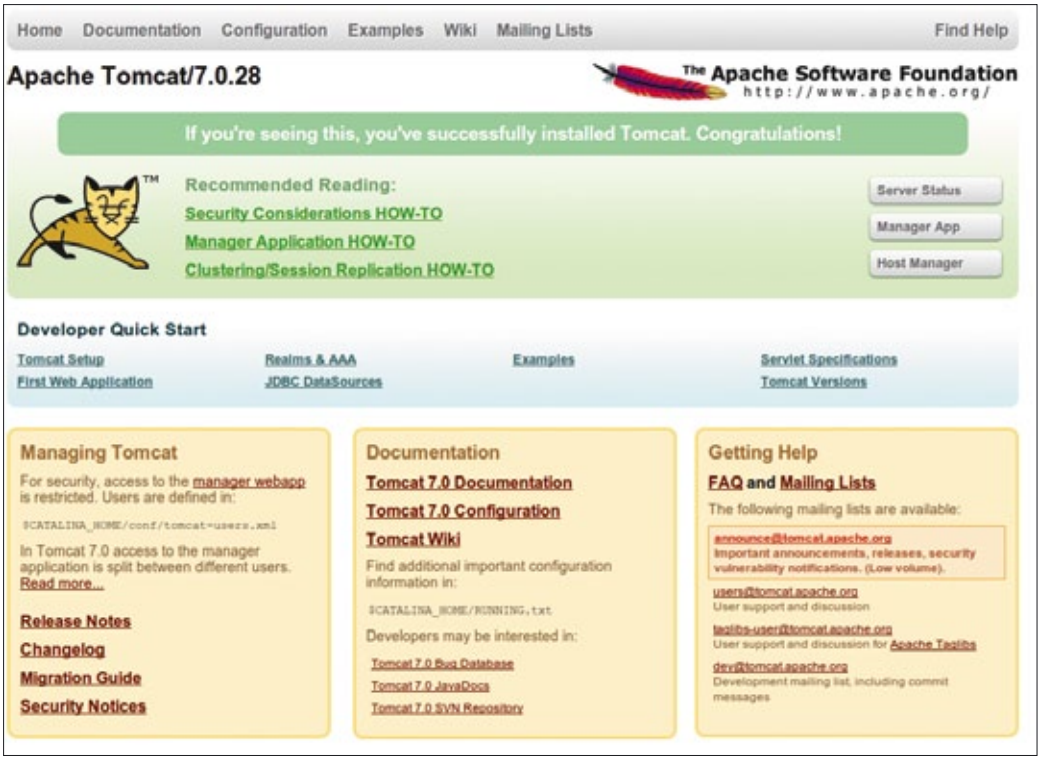


Figure 5. *Apache Tomcat 7 on FreeBSD 9*

content constrained by manual keywords or tagging, but databases, and disparate files themselves can be searched for content (Figure 2). This results in a major leap in functionality and the ability to find exactly what is wanted in an ordered and logical fashion. With the addition of taxonomies and algorithms, search starts to become highly intelligent – for example related content and facets (e.g. colour, price etc.) (Figure 3).

Extending this philosophy further still – what is an internet search engine? Effectively it is the middleware between the user and billions of pages of dispa-

Great Specials

On FreeBSD & PC-BSD Merchandise

Give us a call & ask about our
SOFTWARE BUNDLES
1.925.240.6652

$39.95
FreeBSD 9.0 Jewel Case CD Set
or FreeBSD 9.0 DVD

$29.95
PC-BSD 9.0 DVD

$49.95 Save a BUNDLE!
The PC-BSD 9.0 Users Handbook
PC-BSD 9.0 DVD

$99.95
The FreeBSD CD or DVD Bundle

Inside each CD/DVD Bundle, you'll find:
FreeBSD Handbook, 3rd Edition
Users Guide FreeBSD Handbook, 3rd Edition, Admin Guide
FreeBSD 9.0 CD or DVD set
FreeBSD Toolkit DVD

Stylish Dress Attire
Look Your Professional Best

Comfy Hoodies
Stay Warm in Pullovers & Zip Ups

T-Shirts
Lots of Styles to Choose From

FreeBSD 9.0 Jewel Case CD/DVD $39.95

CD Set Contains:

- **Disc 1:** Installation Boot LiveCD (i386)
- **Disc 2:** Essential Packages Xorg, GNOME2 (i386)
- **Disc 3:** Installation Boot LiveCD (amd64)
- **Disc 4:** Essential Packages Xorg, GNOME2 (amd64)

FreeBSD 8.2 CD $39.95
FreeBSD 8.2 DVD $39.95

FreeBSD Subscriptions

Save time and $$$ by subscribing to regular updates of FreeBSD

FreeBSD Subscription, start with CD 9.0 ..... $29.95
FreeBSD Subscription, start with DVD 9.0 ..... $29.95
FreeBSD Subscription, start with CD 8.2 ..... $29.95
FreeBSD Subscription, start with DVD 8.2 ..... $29.95

PC-BSD 9.0 DVD (Isotope Edition)

PC-BSD 9.0 DVD $29.95
PC-BSD Subscription $19.95

The FreeBSD Handbook

The FreeBSD Handbook, Volume 1 (User Guide) $39.95
The FreeBSD Handbook, Volume 2 (Admin Guide) $39.95

The FreeBSD Handbook Specials

The FreeBSD Handbook, Volume 2 (Both Volumes) $59.95
The FreeBSD Handbook, Both Volumes & FreeBSD 9.0 ..... $79.95

PC-BSD 9.0 Users Handbook $24.95

BSD Magazine $11.99

The FreeBSD Toolkit DVD $39.95

FreeBSD Mousepad $10.00

FreeBSD & PCBSD Caps $20.00

BSD Daemon Horns $2.00

Bundle Specials!
Save $$$

Just Plain Fun
Mousepads & Novelty Horns

BSD Magazine
Available Monthly

For even **MORE** items
visit our website today!

www.FreeBSDMall.com

Table 1. *Sole core JAR and source files*

File-name	Description	Download from
apache-nutch-2.0-src.tar.gz	Apache nutch source	http://www.apache.org/dyn/closer.cgi/nutch
apache-solr-4.0.0-BETA.tgz	Apache Solr examples and JAR/WAR files	http://lucene.apache.org/solr
tika-app-1.2.jar	Apache Tika JAR file	http://tika.apache.org/download.html

rate content on millions of web-servers. Of course. if the user is aware of the address of a site they can go directly there with the browser, but more frequently users search for strongly related terms and select a link. The visitor is totally unconcerned about the where or how or the mechanics – the information appears like magic. To the end user, the systems appear unified and integrated even although they are in reality separate.

This approach lends itself well to enterprise integration, but until recently the difficulty has been as always the API's and accessing the internal content of documents reliably. The Systems architect has been dependent on the vendor providing hooks into the legacy system, and often this is very expensive, specialized or limited. However, with Apache Solr, these boundaries can be crossed and intelligent search made available across the enterprise (Figure 4).

So what are Solr, Tika and Nutch?

Apache Solr is an enterprise grade search platform which emerged from the Lucene project. Highly scalable its major features include powerful full-text search, hit highlighting, faceted search, dynamic clustering, database integration, and the ability to index a wide range of documents and meta-data from disparate file formats.

The Apache Tika toolkit detects and extracts meta-data and structured text content from various documents using existing parser libraries.

Apache Nutch is a web crawler used to pull content from websites. Robust and scalable, Nutch can prioritize what pages are fetched first.

The biggest challenge to implementing Solr effectively is designing a suitable schema that is powerful enough to answer queries yet flexible enough to be extensible. Solr is not an RDBMS, it excels at language manipulation, ranking and faceting as well as parsing and extracting content and meta-data from a wide variety of sources when used with Apache Tika and Nutch. This requires a different approach when it comes to system design from that of database and relational architectures.

In this series of articles, we will build a Solr 4 search engine under FreeBSD 9 (clean install) with the latest version of Tomcat 7. We will look at the Solr management interface, indexing some sample documents and designing schema's etc.

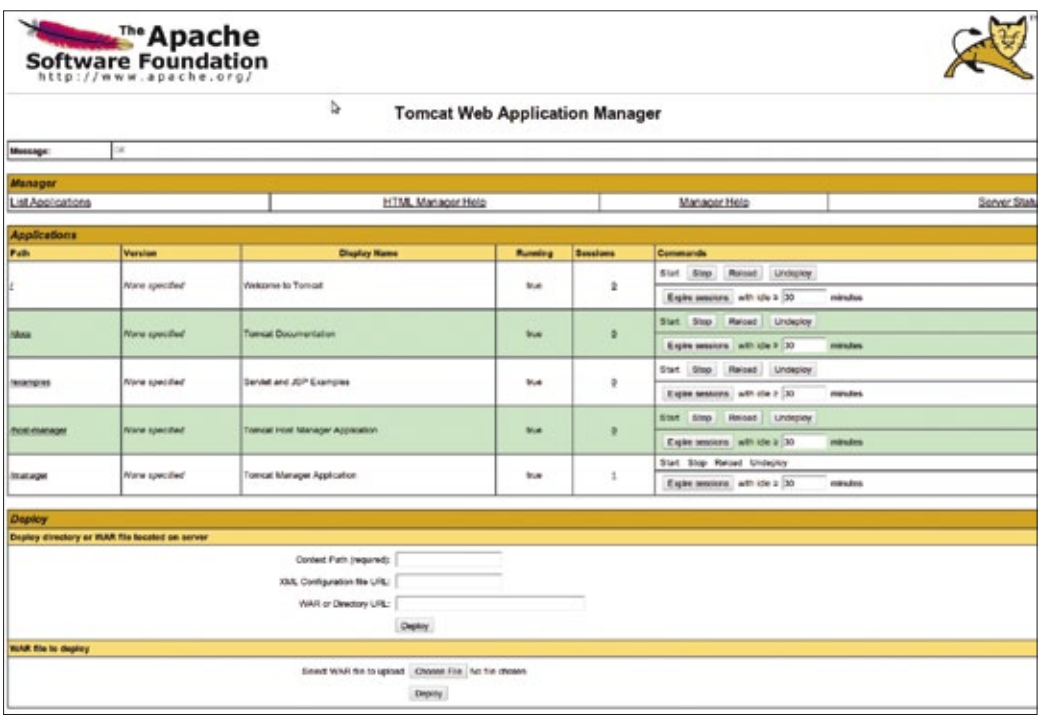


Figure 6. *Tomcat manager after login*

Let's Get Started

First download diablo-caffe-freebsd7-i386-1.6.0_07-b02.tar.bz2 and accept the licence for Diablo Version 1.6.0-0. Place this file in `/usr/ports/distfiles`.

Then download the following files for the Solr install (using the most convenient mirror) and place in temporary directory somewhere (e.g. `/tmp/solr`) (Table 1).

As root, bring the ports tree up to date and install Tomcat 7 from source:

```
portsnap fetch update
cd /usr/ports/www/tomcat7
make install BATCH=YES
ln -s /usr/local/apache-tomcat-7.0/logs /var/log/tomcat7
```

This will download the source code, building Tomcat 7 from scratch and will take some time so go and grab a coffee. When the `BATCH=YES` switch is included to make, it should run unattended with the default settings provided you have Diablo Version 1.6.0-0 in distfiles.

Once complete, add the following line to `/etc/rc.conf` using your favourite editor to ensure Tomcat starts correctly on reboot:

```
tomcat7_enable="YES"
```

If you want to use the Tomcat web manager via a web browser, add the following lines to `/usr/local/apache-tomcat-7.0/conf/tomcat-users.xml` before the `</tomcat-users>` tag (Use a strong password in an production environment):

```
<role rolename="manager-gui"/>
<user username="tomcat" password="tomcat"
                roles="manager-gui"/>
```

Start tomcat:

Listing 1. *Installing Tomcat & Solr*

```
/usr/local/etc/rc.d/tomcat7 stop
cd /tmp/solr
tar xvzf apache-solr-4.0.0-BETA.tgz
cd /tmp/solr/apache-solr-4.0.0-BETA/example
cp -r solr /home
cp -r exampledocs /home/solr
mv /home/solr/bin /home/solr/collection1/bin
mkdir /home/solr/collection1/lib

find /tmp/solr/apache-solr-4.0.0-BETA -iname "*.jar" -exec cp -v {} /home/solr/collection1/lib \;

chown -R www:www /home/solr
cd ../dist
unzip apache-solr-4.0.0-BETA.war -d /usr/local/apache-tomcat-7.0/webapps/solr
chown -R www:www /usr/local/apache-tomcat-7.0/webapps/solr
touch /usr/local/apache-tomcat-7.0/conf/Catalina/localhost/solr.xml
chown www:www /usr/local/apache-tomcat-7.0/conf/Catalina/localhost/solr.xml
```

Listing 2. *Adding correct Java library path to Tomcat solrconfig.xml file*

```
<!--
<lib dir="../dist/" regex="apache-solr-cell-\d.*\.jar" />
<lib dir="../contrib/extraction/lib" regex=".*\.jar" />
<lib dir="../dist/" regex="apache-solr-clustering-\d.*\.jar" />
                <lib dir="../contrib/clustering/lib/" regex=".*\.jar" />
<lib dir="../dist/" regex="apache-solr-langid-\d.*\.jar" />
<lib dir="../contrib/langid/lib/" regex=".*\.jar" />
<lib dir="../dist/" regex="apache-solr-velocity-\d.*\.jar" />
<lib dir="../contrib/velocity/lib" regex=".*\.jar" />
-->

<lib dir="lib" />
```

Figure 7. *Solr dashboard*

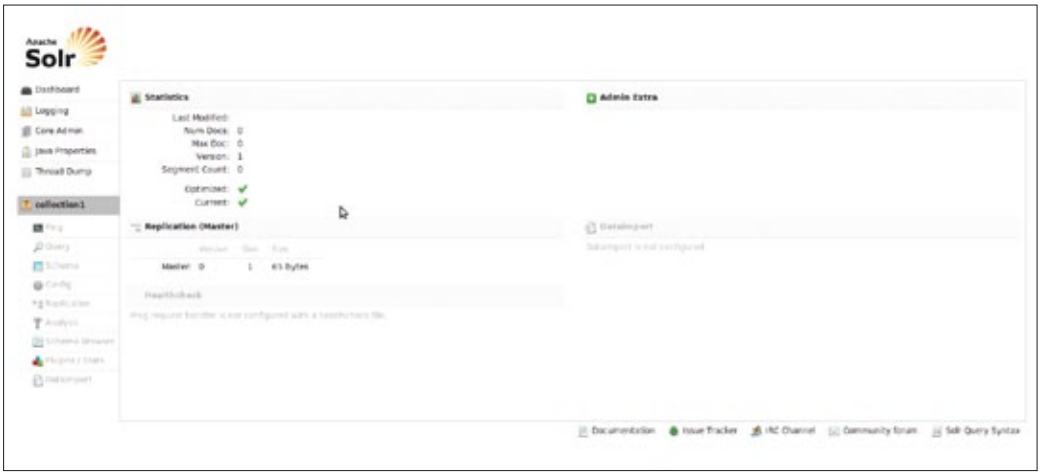


Figure 8. *The example collection1*

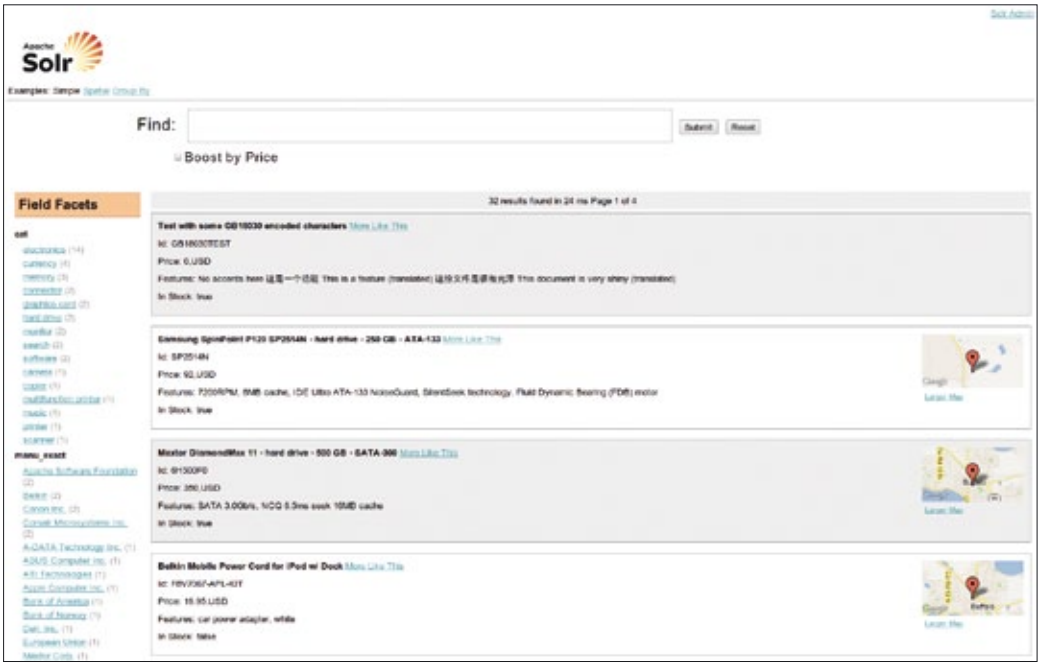


Figure 9. *Solr faceted geo-location search*

Listing 3. *Result returned for "Enterprise"*

```
This XML file does not appear to have any style information
associated with it. The document tree is shown below.

▼<response>
  ▼<lst name="responseHeader">
     <int name="status">0</int>
     <int name="QTime">4</int>
     ▼<lst name="params">
     <str name="q">Enterprise</str>
     <str name="wt">xml</str>
     </lst>
  </lst>
  ▼<result name="response" numFound="1" start="0">
  ▼<doc>
     <str name="id">SOLR1000</str>

     <str name="manu">Apache Software Foundation</str>
     ▼<arr name="cat">
        <str>software</str>
        <str>search</str>
     </arr>
     ▼<arr name="features">
        ▼<str>
           Advanced Full-Text Search Capabilities
                 using Lucene
        </str>
        <str>0ptimized for High Volume Heb Traffic</str>
        <str>Standards Based Open Interfaces - XML and
                 HTTP</str>
        <str>Comprehensive HTML Administration
                 Interfaces</str>
        ▼<str>
        Scalability - Efficient Replication to other
                 Solr Search Servers
        </str>
        ▼<str>
        Flexible and Adaptable with XML configuration
                 and Schema
        </str>
        ▼<str>
        Good Unicode support: héllo (hello with an
                 accent over the e)
        </str>
     </arr>
     <float name="price">0.0</float>
     <str name="price_c">0,USD</str>
     <int name="popularity">10</int>
        <bool name="inStock">true</bool>
        <date name="incubationdate_dt">2006-01-
                 17T00:00:00Z</date> clong name="_
                 version_">1411798689903542272</long>
  </doc>
  </result>
</response>
```

Listing 4. *Search for "Video"*

```
This XML file does not appear to have any style information
associated with it. The document tree is shown below.

▼<response>
  ▼<lst name="responseHeader">
     <int name="status">0</int>
     <int name="QTime">23</int>
     ▼<lst name="params">
        <str name="fl">name,id,score</str>
        <str name="q">video</str>
        </lst>
     </lst>
  ▼<result name="response" numFound="3" start="0"
                 maxScore="0.500039">
    ▼<doc>
     <str name="id">MA147LL/A</str>
     <str name="name">Apple 60 GB iPod with Video
                 Playback Black</str> <float
                 name="score">0.500039</float>
    </doc>

     <str name="id">EN7800GTX/2DHTV/256M</str>
     <str name="name">ASUS Extreme N7800GTX/2DHTV (256
                 MB)</str>
     <float name="score">0.3849302</float>
    </doc>
  ▼<doc>
     <str name="id">100-435805</str>
     <str name="name">ATI Radeon X1900 XTX 512 MB PCIE
                 Video Card</str>
     <float name="score">0.3849302</float>
  </doc>
  </result>
</response>
```

```
/usr/local/etc/rc.d/tomcat7 onestart
```

You should see the following screens at `http://yourserverip:8080` and `http://yourserverip:8080/manager` where "yourserverip" is the external IP address of the your FreeBSD install (Figure 5 and Figure 6).

Installing Solr

Initially, we will run Solr in single core mode without clustering. For this demo we will use the example documents and schemas from Collection1 supplied by Apache (Listing 1).

Edit `/usr/local/apache-tomcat-7.0/conf/Catalina/localhost/solr.xml` as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<Context docBase="./solr" debug="0" crossContext="true">
<Environment name="solr/home" type="java.lang.String"
                 value="/home/solr" override="true"/>
</Context>
```

Edit `/home/solr/collection1/conf/solrconfig.xml` to reflect the following: Listing 2.

Finally, reboot the server to test that everything will come up at boot:

```
reboot
```

Solr should now be running at *http://yourserverip:8080/solr* (Figure 7 and Figure 8).

Indexing and Retrieving Data

```
su
cd /home/solr/exampledocs
pkg_add -r curl
```

Then edit the post.sh file to show:

```
URL=http://localhost:8080/solr/update
```

Index two documents:

```
./post.sh solr.xml monitor.xml
```

You should see the files being posted.

Now visit *http://yourserverip:8080/solr/collection1/select?q=Enterprise&wt=xml.*

You should see your document returned in XML format (Listing 3).

Index all the XML docs in the examples directory:

References and further reading

- Apache Tomcat – *http://tomcat.apache.org/download-70.cgi*
- Apache Solr – *http://lucene.apache.org/solr*
- Apache Tika – *http://tika.apache.org*
- Apache Nutch – *http://nutch.apache.org*

```
./post.sh *.xml
```

Now search for video only displaying the name, id and score field:

```
http://192.168.0.127:8080/solr/collection1/select?q=video&fl=name,id,score
```

You should see 3 results returned in XML format (Listing 4).

Finally, visit *http://192.168.0.127:8080/solr/browse.*

You should see faceting in action. If you encounter fatal tomcat errors (*SEVERE SolrDespatchFilter etc*), check that the *.jar files from contrib and dist trees have been copied across and that `<lib dir />` setting is correct.

In the Next Article ...

We will look at synonyms, stemming and the data handler.

ROB SOMERVILLE

Rob Somerville has been passionate about technology since his early teens. A keen advocate of open systems since the mid eighties, he has worked in many corporate sectors including finance, automotive, airlines, government and media in a variety of roles from technical support, system administrator, developer, systems integrator and IT manager. He has moved on from CP/M and nixie tubes but keeps a soldering iron handy just in case.

EuroBSDcon 2012

19-21 October, Warsaw, Poland

Talks:

Saturday 20th of October

- An Overview of Locking in the FreeBSD Kernel – Kirk McKusick
- Config Management in FreeBSD using Puppet – Edward Tan
- Using routing domains / routing tables in a production network – Peter Hessler
- FreeNAS system architecture – John Hixson
- The pivot_root system call for BSD systems (NetBSD) – Adrian Steinmann
- How to put FreeBSD power into small MIPS switch/router – Aleksandr Rybalko
- Improvements in the IPsec stack and OpenBSD cryptographic framework – Mike Belopuhov
- A call for authentication reform – Dag-Erling Smørgrav
- FreeBSD and NetBSD on APM86290 system on chip – Zbigniew Bodek
- BSD/Unix CLI and TUI Ecology – Andrew Pantyukhin
- OpenBSD's new queueing subsystem – Henning Brauer
- The Warden – FreeBSD and Linux Jail Management – Kris Moore
- Advances in packages and ports in OpenBSD – Marc Espie

Sunday 21st of October

- The BHyVe Hypervisor In Depth – Michael Dexter
- Extension to veriexec which uses digital signatures to verify the provenance of a file – Alistair Crooks
- Tuning ZFS on FreeBSD – Martin Matuska
- Tips on running a conference for 250 people all by yourself – Dan Langille
- Running BSD-licensed Software on BSD-licensed Hardware – Marius Strobl
- OpenBSD and 'real' threads – Philip Guenther
- mplementation of SCTP in Go (FreeBSD) – Olivier Van Acker
- Touch your NetBSD – Pierre Pronchery
- A Fault Aware Global Server Load Balancer in DNS – Stefan D. Caunter, Allan C. Jude
- NetBSD/usermode – Reinoud Zandijk

Talks schedule available on:
http://2012.eurobsdcon.org/agenda/talks/

On 18th and 19th of October tutorials by:

Dru Lavigne, Kirk McKusick, Chris Buechler, Ermal Luci, Radoslaw Kujawa, Tod McQuilin, Peter N. M. Hansteen

Tutorial schedule on: http://2012.eurobsdcon.org/agenda/tutorials/

http://2012.eurobsdcon.org/

Platinium Sponsors

EMC²

systems

WHEEL SYSTEMS

Gold Sponsor

The FreeBSD FOUNDATION

Media

niebezpiecznik.pl

MAGAZINE BSD

PostgreSQL

Partitioning (Part 2)

In the last article data partitioning was introduced and an application example consisting of a forum database was used to explain how to partition tables, migrate data and route queries to the right data set depending on the forum post's "category" and "timing".

What you will learn...

- How to use tablespaces to handle database data
- how to implement partitioning that exploits tablespaces

What you should know...

- basic shell commands
- basic PostgreSQL concepts
- partitioning concepts explained in the previous article

In this paper readers will further extend the application scenario presented in the previous paper, implementing a physical partitioning that keeps tables and data in separate storage devices. All the examples shown here have been tested on a PostgreSQL 9.1 cluster running on a FreeBSD 8.2-RELEASE machine; see the previous article in this series for details about the application scenario and how to reproduce it.

Improving the Partitioning

In the previous article readers saw a simple database, called *forumdb*, populated with around 4 million tuples representing forum posts, contained in a main *thread* table. This table was then partitioned first into a per-category table (e.g., *thread_net*) in order to group posts by their category; subsequently the data was partitioned further based upon the year a post created. Of course, ad-hoc constraint checking as well as triggers and rules were built to route `INSERT` queries and to avoid data corruption (i.e., storing a post into the wrong table).

The situation could be summarized as shown in Listing 1.

While this partitioning is effective, it probably does not achieve the overall goal of allowing for the highest possible performance of an interactive forum. It is worth noting that insertion of new posts will always be performed on the last per-year table of each category;

Listing 1. *The initial situation for the examples*

```
bsdmag=> SELECT relname, reltuples FROM pg_class
WHERE relname like 'thread%' AND relkind = 'r' ORDER BY
                    relname;
       relname         | reltuples
-----------------------+-----------
 thread                |         0
 thread_hw             |         0
 thread_hw_year1991    |     29014
 ...
 thread_hw_year2004    |      5053
 thread_kern           |         0
 thread_kern_year1993  |     43621
 ...
 thread_kern_year2012  |     13255
 thread_misc           |         0
 thread_misc_year1990  |     58282
 ...
 thread_misc_year2012  |     17710
 thread_net            |         0
 thread_net_year1992   |     72943
 ...
 thread_net_year2012   |     22165
```

excluding application bugs and forum recovery, posts that are in the past will not be changed and no post in the past can be added. On the other hand, old posts could be the needed for performing queries, and therefore it is not possible to just discard such posts. This scenario is therefore asymmetric: the last per-year table of each category will be actively used for both queries, updates and insertions, while the other per-year tables will be used exclusively for queries. In such a scenario it is therefore possible to give the "current year" table of each category priority over the other tables, so that queries affecting the current year are optimized in some way to complete faster than queries to other tables. A possible way to achieve this is to use different storage devices for different tables, so that the current-year tables are stored on faster disks while other tables will be stored on slower disks. The feature that allows PostgreSQL to have different storage systems is referred to as tablespaces.

Introduction to Tablespaces

Tablespaces are storage locations, file system hierarchies, that can be used to store database objects (mainly tables and indexes). As explained in the first article of this series, PostgreSQL stores all objects within a file system hierarchy identified by the environment variable `$PGDATA`, in particular the `$PGDATA/base` contains all the databases and their data in files named after the OID of the table/object itself (with a few exceptions).

Tablespaces represent a way to "escape" the `$PGDATA` directory allowing the cluster to use extra disk storage, different speed and architecture disk storage and even different file systems. Several scenarios are possible: not only most frequently accessed tables can be stored on a faster mass storage device, but also indexes can be placed elsewhere. Besides storage speed, tablespaces allows also for a storage capacity scale up.

Box 1. Using a memory disk for experiments

During the writing of this article the author used a memory disk (vnode backed) to simulate a very fast disk attached to the machine and mounted at `/postgresql/fast-disk`. The following are the steps required to reproduce the simulation with a memory disk identified as `/dev/md10`:

```
# touch /postgresql/memory_disk.md
# dd if=/dev/zero of=/postgresql/memory_disk.md bs=1M
                    count=50
# mdconfig -a -t vnode -f /postgresql/memory_disk.md -s
                    50M -u 10
# mdconfig -l -v
md10    vnode       50M  /postgresql/memory_disk.md
# newfs /dev/md10
# mkdir /postgresql/fast-disk
# mount /dev/md10 /postgresql/fast-disk/
```

The usage of a memory disk is beyond of the scope of this article, please refer to the operating system documentation.

Defining a Tablespace

Suppose that the database machine has a fast storage disk mounted at `/postgresql/fast-disk`. In order to make PostgreSQL selectively use such disk a new *tablespace* has to be defined. Defining a tablespace requires the creation of a directory with the right user and permissions so that only PostgreSQL can use it:

```
# mkdir /postgresql/fast-disk/forum_tablespace
# chown pgsql:pgsql /postgresql/fast-disk/forum_tablespace
# chmod 700 /postgresql/fast-disk/forum_tablespace
```

No further disk initialization is required for a tablespace to be used. Having defined the storage location to use as tablespace it is possible to inform PostgreSQL about such location using the `CREATE TABLESPACE` command (as database superuser):

```
forumdb=# CREATE TABLESPACE ts_forum
OWNER forum
LOCATION '/postgresql/fast-disk/forum_tablespace';
```

The tablespace is called `ts_forum` and "points" to the `/postgresql/fast-disk/forum_tablespace directory`; moreover the ownership of this tablespace is granted to the PostgreSQL user `forum`. What happens on disk is that a symbolic link is created from the cluster storage directory to the tablespace location, in particular the `$PGDATA/pg_tblspc` directory contains links to all the tablespaces defined in the cluster:

```
~> ls -l /postgresql/cluster1/pg_tblspc
lrwx------  1 pgsql  pgsql  38 Apr 19 13:49 76871 -> /
                    postgresql/fast-disk/forum_tablespace
```

Thanks to the linking mechanism PostgreSQL can reach all the tablespace storage locations without having to "escaping" from the `$PGDATA` directory.

Having defined the tablespace is now possible to use it to store database objects, in particular tables and their data. The `CREATE TABLE` command has the special option `TABLESPACE` that allows the specification of a tablespace to use; therefore issuing a command like:

```
CREATE TABLE thread_net_2012 ( … ) TABLESPACE ts_forum;
```

Listing 2. *A stored procedure to change tablespace of the current-year tables*

```
CREATE OR REPLACE FUNCTION migrate_tables_to_
                    tablespace()
RETURNS integer
AS
$BODY$
DECLARE
        current_category        category%rowtype;
        current_year            integer;
        migrated_tables         integer;
BEGIN

        migrated_tables := 0;

        SELECT EXTRACT( year FROM current_date )
        INTO   current_year;

        -- iterate over each category
        FOR current_category IN SELECT *
                                FROM category
                                ORDER BY id
                                LOOP

                EXECUTE 'ALTER TABLE '
                        || 'thread_' || current_
                    category.id || '_year' || current_
                    year
                        || ' SET TABLESPACE ts_forum
                    ';

                migrated_tables := migrated_tables +
                    1;

        END LOOP;       -- end of the category
                    iteration

        RETURN migrated_tables;

END;
$BODY$
LANGUAGE plpgsql;
```

will result in the creation of the `thread_net_2012` table that will be stored in `/postrgresql/fast-disk/forum_tablespace` storage hierarchy (`ts_forum` tablespace). Because PostgreSQL allows the storage location of a table to be defined when the table is created, partitioning can be split across different hierarchies. In the above example however the tables are already in place and cannot be re-created. Fortunately, PostgreSQL allows the migration of a table to another tablespace using the `ALTER TABLE SET TABLESPACE` command. It is therefore possible to build a simple stored procedure that iterates over each category and migrates the current year table (see Listing 2). The result will be that each *2012* table (the current year) will have a tablespace while all the others will not:

```
forumdb=> \d thread_net_year2012;
...
Inherits: thread_net
Tablespace: "ts_forum"
```

This means that, on disk, under the tablespace hierarchy there will be a set of files, each one named by its OID (see the first article in this series). Given that the *2012* tables have the following files on disk:

Box 2. How to know which tablespaces are available

It is possible to see which tablespaces are available within a cluster using the special command `\db`, that reports both the location, the tablespace name and the owner of the tablespace itself:

```
forumdb=> \db
                      List of tablespaces
    Name    | Owner |                Location
------------+-------+---------------------------------
 pg_default | pgsql |
 pg_global  | pgsql |
 ts_forum   | forum | /postgresql/fast-disk/forum_
                    tablespace
```

Another way to collect tablespace information is the usage of the pg_tablespace catalog:

```
forumdb=> SELECT spcname, spclocation FROM pg_
                    tablespace;
  spcname   |              spclocation
------------+-----------------------------------------
 pg_default |
 pg_global  |
 ts_forum   | /postgresql/fast-disk/forum_tablespace
```

Listing 3a. *The create_category_tables stored procedure that will exploit the "fast" storage tablespace*

```
CREATE OR REPLACE FUNCTION create_category_tables()
RETURNS integer
AS
$BODY$
DECLARE
        current_category        category%rowtype;
        created_tables          integer;
        current_table_name      text;
        current_year            integer;
        current_query           text;
        current_year_to_check   integer;
        current_max_year        integer;
BEGIN

        created_tables := 0;
        -- iterate over each category
        FOR current_category IN SELECT *
                                FROM category
                                ORDER BY id
                                LOOP


                -- build a dynamic query for creating the
                    table
                --EXECUTE 'DROP TABLE thread_' ||
                    current_category.id;
                EXECUTE 'CREATE TABLE IF NOT EXISTS
                    thread_' || current_category.id
                        || ' ( CHECK(category_pk = ' ||
                    current_category.pk || '), '
                        || ' PRIMARY KEY(pk), '
                        || ' FOREIGN KEY(category_pk)
                    REFERENCES category(pk), '
                        || ' FOREIGN KEY(author_pk)
                    REFERENCES author(pk), '
                        || ' UNIQUE(tid, mid) '
                        || ') INHERITS (thread);';

                created_tables := created_tables + 1;

                -- compute the current year
                current_year := EXTRACT(year FROM
                    current_category.since);
                SELECT EXTRACT( year FROM current_date )
                INTO   current_max_year;

RAISE LOG 'Generating time tables from
    year % to year %', current_year,

WHILE current_year <= current_max_year
    LOOP
      RAISE LOG 'Creating sub-table for
    year %', current_year;
      current_year_to_check := EXTRACT(
    year FROM current_category.since ) +
    current_year - 1;


SELECT 'CREATE TABLE IF NOT EXISTS
    thread_'
        || current_category.id || '_
    year' || current_year
        || ' ( '
        || '   CHECK( '
        || ' EXTRACT(year FROM
    published_on)  = '
        || current_year
        || ') ,'
        || ' PRIMARY KEY(pk), '
        || ' FOREIGN KEY(category_pk)
    REFERENCES category(pk), '
        || ' FOREIGN KEY(author_pk)
    REFERENCES author(pk), '
        || ' UNIQUE(tid, mid) '
        || ') INHERITS ('
        || 'thread_' || current_
    category.id

        || ')'
        INTO current_query;

        -- is this year the current one?
    Do we have to use
        -- the fast tablespace?
        IF current_year = current_max_
    year THEN
           current_query := current_
    query || ' TABLESPACE ts_forum';
        END IF;

        current_query := current_query
    || ';';
        EXECUTE current_query;

       current_year := current_year + 1;
```

Listing 3b. *The create_category_tables stored procedure that will exploit the "fast" storage tablespace*

```
                END LOOP;  -- end of the per-year-while

        END LOOP;       -- end of the category iteration

        RETURN created_tables;

END;
$BODY$
LANGUAGE plpgsql;
```

Listing 4. *A stored procedure that migrates indexes by their names*

```
CREATE OR REPLACE FUNCTION migrate_indexes_to_
                    tablespace()
RETURNS integer
AS
$BODY$

        current_category        category%rowtype;
        current_year            integer;
        migrated_indexes        integer;
BEGIN

        migrated_indexes := 0;

        SELECT EXTRACT( year FROM current_date )
        INTO   current_year;

        -- iterate over each category
        FOR current_category IN SELECT *
                                FROM category
                                ORDER BY id
                                LOOP

                EXECUTE 'ALTER INDEX '
                        || 'thread_' || current_
                    category.id
                        || '_year'   || current_year
                        || '_pkey'
                        || ' SET TABLESPACE ts_forum ';

                EXECUTE 'ALTER INDEX '
                        || 'thread_' || current_
                    category.id
                        || '_year'   || current_year
                        || '_tid_mid_key'
                        || ' SET TABLESPACE ts_forum ';


                migrated_indexes := migrated_indexes +
                    2;

        END LOOP;       -- end of the category iteration

        RETURN migrated_indexes;

END;
$BODY$
LANGUAGE plpgsql;
```

Box 3. How to quickly set up (again) the database

To build up the database as in previous article, and in order to repeat the examples shown here, it is possible to issue the following commands (being connected to the forumdb):

```
DROP TABLE IF EXISTS thread_net CASCADE;
DROP TABLE IF EXISTS thread_misc CASCADE;
DROP TABLE IF EXISTS thread_kern CASCADE;
DROP TABLE IF EXISTS thread_misc CASCADE;
DROP TABLE IF EXISTS thread CASCADE;
DROP TABLE IF EXISTS author;
DROP TABLE IF EXISTS category;
DROP VIEW IF EXISTS vw_thread;
\i 01-forum-database-initial-setup.sql
\i 02-function-populate.sql
SELECT populate_forum();
\i 03-function-create-category-tables.sql
SELECT create_category_tables();
\i 04-migration.sql
SELECT migrate_threads();
\i 05-thread-table-rules.sql
SELECT create_category_rules();
\i 07-partitioning-time.sql
SELECT migrate_threads_by_category_and_time();
SELECT create_category_time_triggers();
VACUUM FULL ANALYZE;
```

Please note that all the scripts come from the GitHub repository (see the references) and are contained in the `bsdmag/05-partitioning` directory. The whole process could require more than 20 minutes to complete, depending on the speed of the hardare.

On The Web

- PostgreSQL official Web Site: *http://www.postgresql.org*
- ITPUG official Web Site: *http://www.itpug.org*
- PostrgeSQL Table Inheritance Documentation: *http://www.postgresql.org/docs/current/static/ddl-inherit.html*
- PostgreSQL Tablespace Documentation: *http://www.postgresql.org/docs/current/static/manage-ag-tablespaces.html*
- GitHub Repository containing the source code of the examples: *https://github.com/fluca1978/fluca-pg-utils*

```
forumdb=> SELECT relname, relfilenode
FROM pg_class WHERE relkind='r' AND relname like
'thread_%_year2012';
       relname        | relfilenode
----------------------+-------------
 thread_hw_year2012   |       89461
 thread_kern_year2012 |       89464
 thread_misc_year2012 |       89467
 thread_net_year2012  |       89470
```

the tablespace hierarchy will contain files with the same names.

In the case where data partitioning has not yet been completed, and the per-year tables have therefore not yet been created, it is possible to change the stored procedure which creates the tables (see Listing 3) so that when the table being created is that of the current year, the "fast" storage will be used.

A tablespace can be used also for storing indexes, thereby improving speed of indexed access to the data. Since the example tables all have only two indexes (the primary key index and a unique index on the couple `tid`, `mid`) it is possible to build a stored procedure that will also migrate the indexes to the new tablespace using the ALTER INDEX SET TABLESPACE statement (see Listing 4).

LUCA FERRARI

Luca Ferrari lives in Italy with his wife and son. He is an Adjunct Professor at Nipissing University, Canada, a co-founder and the vice-president of the Italian PostgreSQL Users' Group (ITPUG). He simply loves the Open Source culture and refuses to log-in to non-Unix systems. He can be reached on line at http://fluca1978.blogspot.com.

Hardening FreeBSD

with TrustedBSD and Mandatory Access Controls (MAC) Part 3

Most system administrators understand the need to lock down permissions for files and applications. In addition to these configuration options on FreeBSD, there are features provided by TrustedBSD that add additional layers of specific security controls to fine tune the operating system for multilevel security.

What you will learn…

- Configuration of the mac_bsdextended module.
- How to use the ugidfw utility

What you should know…

- Basic FreeBSD knowledge to navigate the command line
- Familiarity with loader.conf to enable kernel modules at boot

Since version 5.0 of FreeBSD, the TrustedBSD extensions have been included with the default install of the operating system. By default, this functionality is disabled and requires support to be compiled in or kernel modules to be loaded at boot time. For the purpose of this article, support will be loaded in with kernel modules already available with FreeBSD 9. Part 3 of the TrustedBSD series will cover the basic configuration of the `mac_bsdextended` module.

Warning

Incorrect MAC settings can cause even the root user to not be able to login to the system. Be sure to run these tests on a VM or test machine to avoid any issues with production systems. This article assumes that a fresh install of FreeBSD 9.0 with a separate file system called “data” has been performed before continuing.

As in the previous articles, a certain set of users will help to illustrate how to use mandatory access controls (MAC) to fine tune access to specific file system objects. Listing 1 shows the layout of the users and groups setup on a separate file system called “data” and how to create them. There is a project to enable discretionary files but for this article the focus will be on file system restrictions.

The `mac_bsdextended` module creates essentially a file system firewall that has a syntax similar to the `ipfw` firewall. In order to load the module on boot, add the following to `/boot/loader.conf` as detailed in Listing 2.

Once the system is rebooted, the `ugidfw` utility will be able to make changes using the loaded module. Listing 3 shows the default output from using the `ugidfw` utility which should not list any rules. The sysctl value should show that `mac_bsdextended` is enabled.

Unlike the previous modules, `mac_bsdextended` does not require changes to policy labels to enforce the access controls. Everything is configured using the `ugidfw` utility with the rules being evaluated in order. This utility highlights the ability to restrict access to objects to authorized subjects, which is an important part of mandatory access controls. For this example, user2-reg directory will be changed so that only user2 has access to the directory for which user1 would normally have access through group permissions. Listing 4 shows the usage of `ugidfw`, with the output from user1 trying to access the directory before and after the change.

With the group permissions allowing user1 to access the directory, setting a rule to only allow a user with a uid matching the directory ownership overrides the standard group permissions. Listing 5 shows an additional rule to examine the gid of the subject. However, because of the previous uid rule, the new gid rule is not evaluated.

In order to open up the permissions to any member of the user-reg group, the rule order must be changed. List-

Listing 1. *Directory setup on FreeBSD for several users called /data*

```
# mkdir -p /data/user1-reg
# mkdir -p /data/user2-reg
# touch /data/user1-reg/secret-order.txt
# touch /data/user2-reg/secret-order.txt
# pw user add -n user1 -s /bin/csh -m
# pw user add -n user2 -s /bin/csh -m
# pw group add user-reg -M user1,user2
# passwd user1
Changing local password for user1
New Password:
Retype New Password:
# passwd user2
Changing local password for user2
New Password:
Retype New Password:
# chmod -R 770 /data/user1-reg/ /data/user2-reg
# chown -R user1:user-reg /data/user1-reg
# chown -R user2:user-reg /data/user2-reg
# ls -ltra
total 20
drwxrwxr-x   2 root   operator   512 Sep  2 12:40 .snap
drwxr-xr-x  21 root   wheel     1024 Sep  2 12:40 ..
drwxrwx---   2 user1  user-reg   512 Sep  2 12:58 user1-
                    reg
drwxrwx---   2 user2  user-reg   512 Sep  2 12:58 user2-
                    reg
drwxr-xr-x   5 root   wheel      512 Sep  2 12:58 .
# groups user1
user1 user-reg
# groups user2
user2 user-reg
```

Listing 2. *Loading the mac_biba module on system startup*

```
# echo 'mac_bsdextended_load="YES"' >> /boot/loader.conf
# reboot
```

Listing *3. Output from ugidfw with validation the module is loaded*

```
# sysctl -a security.mac.bsdextended.enabled
security.mac.bsdextended.enabled: 1
# ugidfw
usage: ugidfw add [subject [not] [uid uid] [gid gid]]
                  [object [not] [uid uid] \
    [gid gid]] mode arswxn
       ugidfw list
       ugidfw set rulenum [subject [not] [uid uid] [gid
                          gid]] [object [not] \
    [uid uid] [gid gid]] mode arswxn
       ugidfw remove rulenum
# ugidfw list
0 slots, 0 rules
```

Listing *4. Using ugidfw to restrict access to the the user2-reg directory*

```
# echo "TooManySecrets!" > /data/user2-reg/secret-order.
                    txt
# su - user1
%cd /data/user2-reg/
%cat secret-order.txt
TooManySecrets!
%exit
logout
# ugidfw set 1 subject uid user1:user2 object uid
                    user1:user2 filesys /data ! uid_of_
                    subject mode n
# su - user1
%cd /data
%ls -ltra
ls: user2-reg: Permission denied
total 16
drwxrwxr-x   2 root   operator   512 Sep  2 12:40 .snap
drwxr-xr-x  21 root   wheel     1024 Sep  2 12:40 ..
drwxr-xr-x   5 root   wheel      512 Sep  2 12:58 .
drwxrwx---   2 user1  user-reg   512 Sep  2 13:25 user1-
                    reg
```

Listing 5. *Using gid to allow access to the object. Rule 1 triggers for user1 when trying to view the user2-reg directory and vice versa with user2 trying to view user1-reg*

```
# ugidfw set 2 subject uid user1:user2 object uid
                    user1:user2 filesys /data ! gid_of_
                    subject mode n
# ugidfw list
3 slots, 2 rules
1 subject uid user1:user2 object uid user1:user2 filesys
                    /data ! uid_of_subject mode n
2 subject uid user1:user2 object uid user1:user2 filesys
                    /data ! gid_of_subject mode n
# su - user1
%cd /data
%ls -ltra
ls: user2-reg: Permission denied
total 16
drwxrwxr-x   2 root   operator   512 Sep  2 12:40 .snap
drwxr-xr-x  21 root   wheel     1024 Sep  2 12:40 ..
drwxr-xr-x   5 root   wheel      512 Sep  2 12:58 .
drwxrwx---   2 user1  user-reg   512 Sep  2 13:25 user1-reg
%exit
logout
# su - user2
%cd /data
%ls -ltra
ls: user1-reg: Permission denied
total 16
drwxrwxr-x   2 root   operator   512 Sep  2 12:40 .snap
drwxr-xr-x  21 root   wheel     1024 Sep  2 12:40 ..
drwxr-xr-x   5 root   wheel      512 Sep  2 12:58 .
drwxrwx---   2 user2  user-reg   512 Sep  2 13:26 user2-
                    reg
%
```

Listing 6. *Rule for allowing user1 and user2 to access anything with the gid of user-reg*

```
# ugidfw set 1 subject uid user1:user2 object uid
                    user1:user2 filesys /data ! gid_of_
                    subject mode n

3 slots, 1 rules
1 subject uid user1:user2 object uid user1:user2 filesys
                    /data ! gid_of_subject mode n
# su - user1
%cd /data/user2-reg/
%cat secret-order.txt
TooManySecrets!
%
```

ing 6 shows the rule to allow user1 and user2 to access a directory if their group id matches.

The examples in this article used directories as an easy way to highlight the usage of the `ugidfw` utility and the `mac_bsdextended` module. Moving beyond this example, the permissions could be extended to go beyond the data file system and to all file systems to restrict user1 and user2 across the operating system. These controls allow for an additional layer of security in the case that a user creates a file or directory that does not restrict access. With the uid being set to that of the user with the file system firewall, access restrictions can be uniform so that the individual user must give access to the file. In later articles, the MAC modules will be combined to present different layers of security and to help with classifying information.

References

- FreeBSD Handbook – Mandatory Access Cotnrol: *http://www.freebsd.org/doc/handbook/mac.html*
- MAC bsdextended Module: *http://www.freebsd.org/doc/handbook/mac-bsdextended.html*
- Mandatory Access Control: *http://en.wikipedia.org/wiki/Mandatory_access_control*
- ugidfw: *http://www.freebsd.org/cgi/man.cgi?query=ugidfw&sektion=8*
- ugidfw tutorial: *http://www.screamingelectron.org/forum/showthread.php?t=2809*
- TrustedBSD: *http://www.trustedbsd.org/*

MICHAEL SHIRK

Michael Shirk is a BSD zealot who has worked with OpenBSD and FreeBSD for over 6 years. He works in the security community and supports Open-Source security products that run on BSD operating systems.

Interview with

Jeroen van Nieuwenhuizen

Jeroen van Nieuwenhuizen was the chair of the EuroBSDcon 2011 organizing committee. Currently, he is one of the members of the EuroBSDcon Foundation board. He came in contact with Unix in 1997 and started to work with the BSDs in 2002. In his daily life Jeroen works as a Unix Consultant for Snow B.V.

Can you tell us what the EuroBSDcon Foundation is about?

Jeroen van Nieuwenhuizen: The EuroBSDcon Foundation is an idea that existed for several years already. After the 2011 conference, Fred Donck, Paul Schenkeveld and I decided we should go the extra mile and realize it.

What is the mission of the EuroBSDcon Foundation?

JvN: The goal of the EuroBSDcon Foundation is to make it easier to hand over experience between years.

Also, financial resources can be transferred from one conference to another. If one conference has money left at the end, it can be transferred to next year's conference to cover some potential future loss. Additionally, the Foundation can also help with infrastructure when necessary. For example, the EuroBSDcon Foundation is handling the registration for the EuroBSDcon 2012.

Where did the idea of a EuroBSDcon Foundation come from? What made 2011 the year you went the extra mile? What happened?

JvN: I don't know exactly who first came up with the idea of the Foundation, because the idea was around before I got involved. In Karlsruhe in 2010 the idea was put back on the agenda and the idea was to have the Foundation ready to support the organization of EuroBSDcon 2011. However, due to the amount of work to setup a foundation and all the legal issues involved it proved too difficult to get it up and running in time. During and after the EuroBSDcon 2011 we realized we were facing the same problems as earlier years and decided to move forward.

How do they pass knowledge from one event to the other? Wiki? White papers or something else?

JvN: One of the issues we ran into this year was that the Foundation started after the 2012 organizers and a lot of the 2011 experience isn't documented in a format that would be directly useful for other organizers. So currently most knowledge is passed by email, IRC and phone.

I like to look at 2012 as an 'experimental' year of how local organization and the Foundation should work together. Some improvements that we can make are better templates for budgeting, sponsor benefits and the overall planning.

Furthermore, we are working to get infrastructure, like a wiki and version management, in place.

Who are the members of the EuroBSDcon Foundation board?

JvN: The EuroBSDcon Foundation board currently has 8 members. Erwin Lansing (member of the FreeBSD Foundation), S. P. Zeidler (member of the NetBSD Foundation), Henning Brauer (representing the OpenBSD com-

munity), Pawel Jakub Dawidek (organizer of EuroBSDcon 2012), Mitja Muženič (candidate for EuroBSDcon 2013) and Fred Donck, Paul Schenkeveld and me (organizers of EuroBSDcon 2011).

By organizing the board this way we hope to make sure each BSD project is treated equally and that experience is passed on between years.

How can people help the EuroBSDcon Foundation?

JvN: We are, of course, always looking for (international) sponsor contacts. Having a network of potential sponsors for the coming EuroBSD conferences would make it easier to organize them in the future. It can also give sponsors more visibility during the year, e.g. their logo can be on the website when it is announced instead of being visible a few months before the conference itself.

Another way people might help is by donating some of their spare time. For example by volunteering to help design the website for the EuroBSDcon Foundation.

How can our readers get involved if they want to donate time?

JvN: A few things come to mind. We would like to see someone with great graphical skills to help design a logo for the Foundation and when needed by local organizers for that year's EuroBSDcon.

Another area we could use help is with marketing. Most non-local promotion is now done via the mailing lists and BSDmag. It would be great if we could have someone promote EuroBSDcon in every country of Europe and reach out to more BSD enthusiasts. And of course having someone to look into organizing the EuroBSDcon in their own country.

Contact us with ideas and input by sending an email to *info@eurobsdconfoundation.org*.

Are there any particular skills they should have to be able to help? Are there any requirements?

JvN: The most important qualities are being enthusiastic about BSD and willing to learn with us.

What do volunteers gain in exchange? I'm sure that things like fun and experience comes first, but maybe also some good connections or references that can be helpful as well?

JvN: Helping with the EuroBSDcon indeed helps you gain experience. One of the important thing you learn is to work and communicate with people from different cultures and countries. This is a plus, because having good communication skills is a huge advantage when looking for a job in the IT industry.

How do you select the topic and speakers for the conference? Is there a chance for young geeks as well? Are there any requirements that needs to be met in order to be accepted as a speaker?

JvN: The tutorials and talks are selected by the program committee, so this is not directly decided by the EuroBSDcon Foundation. The main criteria are the quality of the talk or tutorial proposal and the room available in the schedule. In an ideal situation the best talks and tutorials are selected while maintaining a balance between talks about the different BSDs. In reality, this might be a little harder to realize due to budget constraints and the prices of airplane tickets. For example sometimes a choice has to be made to have one superb talk from Australia or 4 very good talks from different european countries.

Due to the focus on the quality, I would say the chances for young geeks are as good as for the most seasoned speakers. Therefore, I would like to ask young geeks not to hesitate about sending in their talk proposals, because having more choice among talks and getting more people involved can only benefit the BSDs.

We are also looking into how to improve the selection criteria and make them more transparent. As a result, we are planning a review of the selection process and how we can improve it shortly after EuroBSDcon 2012.

When do you start planning for the next conference? Just after one is finished? What is involved in the planning?

JvN: Ideally we announce the location of the next EuroBSDcon at the closing session of the current. The idea we have to make this attainable is to have 2 candidate organizers for the next year as a member of the foundation board during this year. That way the candidates can see how organizing a EuroBSDcon works and make an informed decision whether they will be able to host the next EuroBSDcon.

Is EuroBSDcon Foundation going to be a "possible contact" for the BSD community and lovers?

JvN: Currently the main role we can provide in that regard is to be a known contact point for questions regarding the EuroBSDcon during the years to come. Especially we think this might make it much easier for international sponsors to keep being a sponsor, without the hassle trying to find the contact for the next EuroBSDcon.

Have you considered that such a foundation can have a more "political" goal? I mean, like creating a "BSD lobbying group" in Europe, as it is with OS for example.

JvN: I think it is too early to focus on things like that. The main focus we have is to make the organization of the EuroBSDcon easier. If we start to focus on too many goals, we might not reach our main goal.

Are you planning to extend the activity of the Foundation or will it always be dedicated to only support EuroBSDcon? What are your plans for the future? Where are you heading?

JvN: Our current and primary focus is the EuroBSDcon. One of the things we discussed is that we are willing to provide support to other BSD conferences. For example by providing our help with the registration system, which has been rewritten and can now easily support more than one conference.

Are there any particular flavors of BSD you prefer and why?

JvN: All the BSDs have their strong points. Therefore we have a representation of the different major BSDs on the board to ensure that a fair balance between the BSDs, regarding the EuroBSDcon, is kept.

Looking at my own situation, I am an advocate of using what is useful in your particular situation and of learning from each other. For example I am mainly using FreeBSD myself because the jail architecture solves the needs I currently have. I would, however, not be able to login to them securely without OpenSSH from the OpenBSD project. And without Jörg Sonnenberger's talk about how NetBSD started using fossil as version management system I would not have solved my version management needs as I have now. I am also very interested in the way DragonFly BSD is going with the HAMMER file system, although I still have to look into it more. I also liked the Minix3 (although not a BSD) talk last year and how they are making their OS very robust against failure.

What are your opinions on how BSD is developing? What improvements, if any, would you like to see?

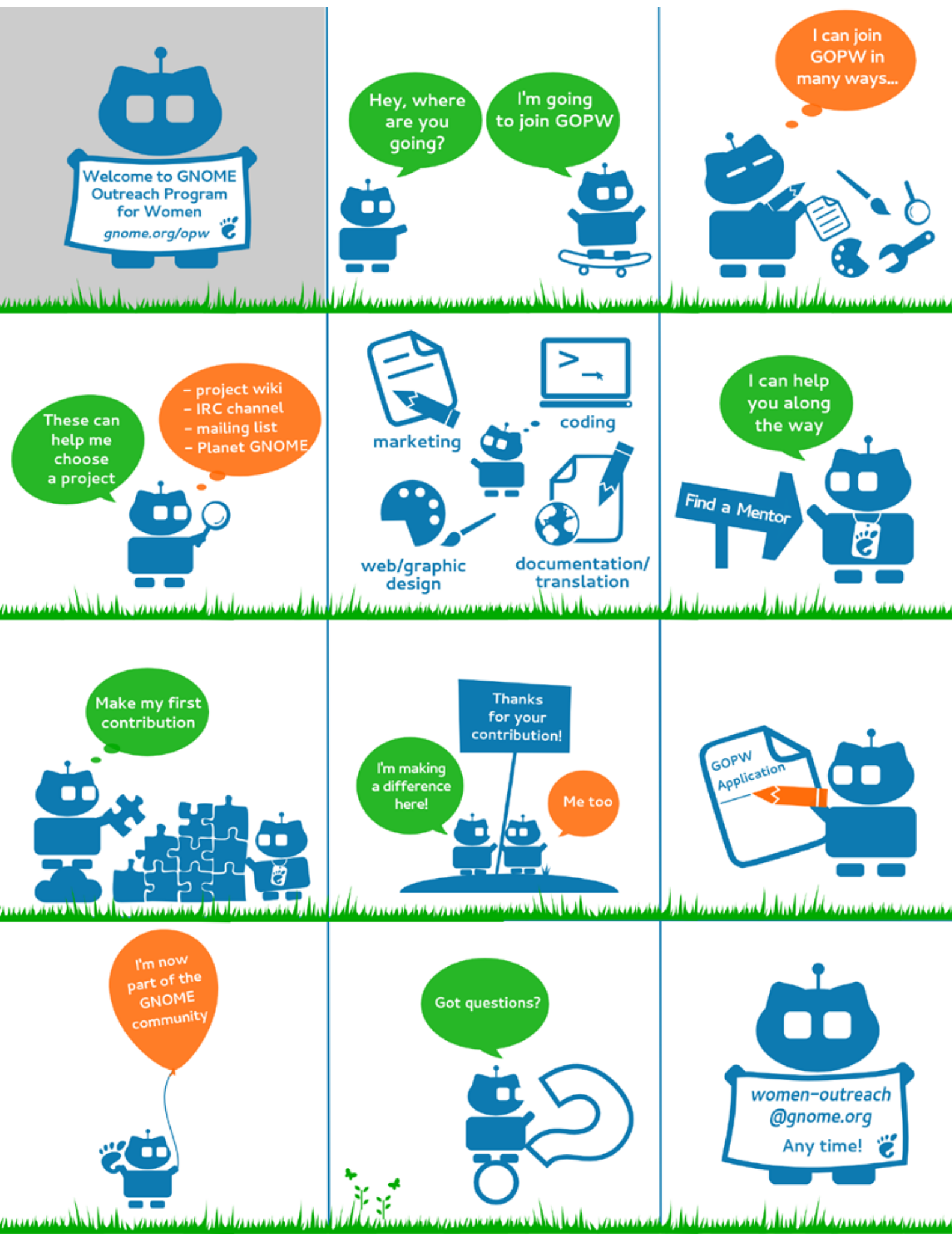
JvN: Technically the BSDs are very strong. One of the things that differentiates them from other open source operating systems is that good design is put before dirty hacks, which makes them very reliable. Being reliable and stable, however, does not make you the most popular. Being relatively unknown is a disadvantage when suggesting to non-BSD aware managers that BSD might solve a particular problem. And BSD has some other problems in this non-technical area which are hard to address.

Looking at the technical side, a kickstart or autoyast like installation infrastructure might be a huge win for mass installations. One idea that comes to mind is BSD support in spacewalk (*http://spacewalk.redhat.com*). Improvements in HA capabilities might also be a win.

So the EuroBSDcon Foundation is involved in organizing the 2012 conference?

JvN: We are supporting the 2012 organization. The organizing it self is done by Pawel Jakub Dawidek and his team in Poland. One of the points we want to keep is that each country organizes the EuroBSDcon according to their own ideas. So for 2012 the EuroBSDcon Foundation is providing help with handling international sponsors and setting up the registration system. Especially the last part proved to be difficult in earlier years. This year, we have rewritten the registration system to become more generic so we won't have problems with that in the upcoming years.

by BSD Team


Welcome to GNOME Outreach Program for Women
gnome.org/opw
Hey, where are you going?
I'm going to join GOPW
I can join GOPW in many ways...
These can help me choose a project
- project wiki
- IRC channel
- mailing list
- Planet GNOME
marketing
coding
web/graphic design
documentation/ translation
I can help you along the way
Find a Mentor
Make my first contribution
Thanks for your contribution!
I'm making a difference here!
Me too
GOPW Application
I'm now part of the GNOME community
Got questions?
women-outreach @gnome.org
Any time!

meet BSD

California 2012

Register Today!

November 3rd & 4th

The event is being held at YAHOO! in Sunnyvale, CA

Registration is available at www.MeetBSD.com/registration

MeetBSD California 2012 promises to be an experience unlike any other.

MeetBSD California is not your average conference - it's a meeting of the minds from all over the BSD community. MeetBSD California 2012 will feature community - driven break - out sessions, discussion groups, and 5 -10 minute "lightning talks," as well as longer talks from seasoned BSD experts.

@MeetBSDCA

www.facebook.com/MeetBSDCalifornia